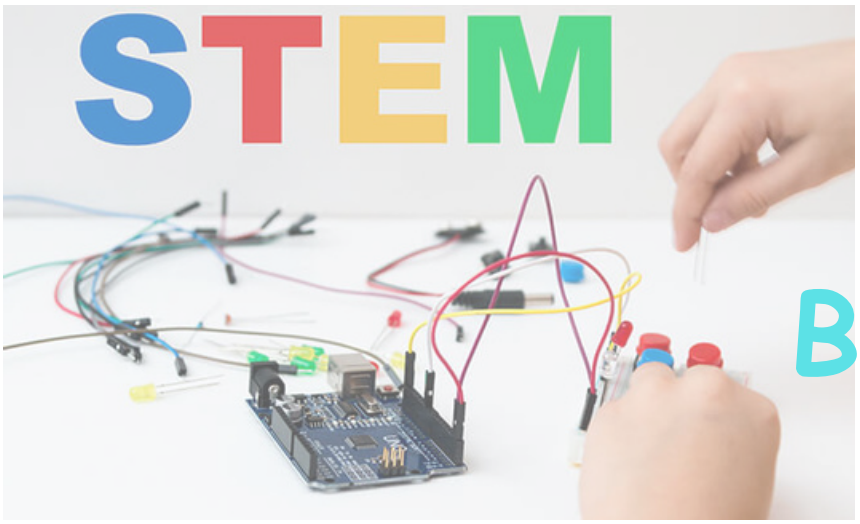
FROM GAME TO STEM

Co-funded by the
Erasmus+ Programme
of the European Union

Lesson plans for STEM subjects with Arduino

Math

Physics



Biology

Chemistry

Ecology

English

FROM GAME TO STEM

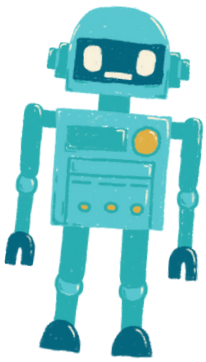
With the support of the Erasmus+ Programme of the European Union

Lesson 1
Arduino Math Lesson

Trigonometry with Arduino

Duration:

3 periods



English

Objective:

- Students will understand basic trigonometric concepts, including sine, cosine, and tangent.
- Students will apply trigonometric functions to solve real-world problems.
- Students will program an Arduino to create a simple angle measuring device using a servo motor.

Materials:

- Arduino boards (one per student or group)
- Servo motors
- Breadboards
- Jumper wires
- Protractors
- Rulers
- USB cables for programming
- Computers with Arduino IDE installed
- Projector or whiteboard for demonstrations

Activities

Day 1

<u>Introduction to Trigonometry (15 minutes):</u>

- Begin the lesson by introducing the concept of trigonometry and its importance in real-world applications.
- Briefly explain sine, cosine, and tangent as ratios of sides in right triangles.
- Discuss how trigonometric functions can be used to solve problems involving angles and distances.

<u>Trigonometry Basics (30 minutes):</u>

- Dive deeper into the definitions of sine, cosine, and tangent.
- Provide examples of how these functions are used to find missing angles or side lengths in right triangles.
- Allow students to practice solving basic trigonometry problems on paper.

<u>Introduction to Arduino and Servo Motors (20 minutes):</u>

- Introduce Arduino as a platform for creating gadgets and explain its components (microcontroller, sensors, actuators).
- Explain the concept of servo motors and how they can be controlled to move to specific angles.
- Show examples of servo motor control with Arduino.

<u>Hands-on Activity (45 minutes):</u>

- Divide students into pairs or small groups.
- Provide each group with an Arduino board, a servo motor, a breadboard, and jumper wires.
- Instruct students to assemble a simple servo motor angle measuring device using a protractor as a reference.
- Guide students in writing Arduino code to control the servo motor to move to a specified angle based on user input.

Day 2

Review of Trigonometry Concepts (20 minutes):

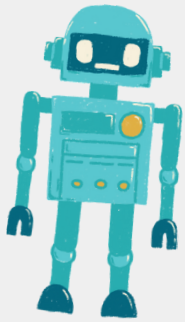
- Begin the second day by reviewing the sine, cosine, and tangent concepts learned on Day 1.
- Provide additional practice problems for students to solve using trigonometric functions.

Programming the Angle Measuring Device (45 minutes):

- Continue with the hands-on activity from Day 1.
- Instruct students to complete their Arduino code to accurately measure and display angles using the servo motor and a numerical display (e.g., Serial Monitor).
- Encourage students to apply trigonometry to convert the servo's position into angles.

Here's a sample Arduino code for creating a simple angle measuring device using a servo motor. This code allows the servo motor to move to a specified angle based on user input and displays the measured angle on the Serial Monitor.

```
#include <Servo.h>
Servo myservo; // Create a Servo object
void setup() {
myservo.attach(9); // Attaches the servo to digital pin 9
Serial.begin(9600); // Initialize serial communication for debugging
}
void loop() {
int angle; // Variable to store the desired angle
Serial.println("Enter an angle (0-180): ");
while (!Serial.available()) {
  // Wait for user input
}
angle = Serial.parseInt(); // Read the angle entered by the user
if (angle >= 0 && angle <= 180) {
// Check if the entered angle is within the valid range
myservo.write(angle); // Move the servo to the specified angle
delay(500); // Delay for stability
Serial.print("Measured angle: ");
Serial.print(angle);
Serial.println(" degrees");
} else {
Serial.println("Invalid angle. Please enter a value between 0 and 180.");
}
}
```



In this code:

We include the Servo library to control the servo motor.
In the setup() function, we attach the servo to digital pin 9 and initialize serial communication for debugging.
In the loop() function, we read the user's input for the desired angle using the Serial Monitor. The user is prompted to enter an angle between 0 and 180 degrees.
We check if the entered angle is within the valid range (0 to 180 degrees). If it is, we move the servo to the specified angle using myservo.write(angle) and display the measured angle on the Serial Monitor.
If the entered angle is outside the valid range, we display an error message.
To use this code with your students, make sure they connect the servo motor to digital pin 9 on the Arduino and open the Serial Monitor to input angles and observe the measured angles.

Day 3

Project Presentation and Discussion (60 minutes):

- Each group presents their Arduino angle measuring device to the class.
- Encourage students to explain how they applied trigonometry concepts in their projects.
- Discuss the real-world applications of angle measuring devices and trigonometry.
- Facilitate a class discussion on the challenges and solutions encountered during the project.

Assessments

Assess students based on their participation, code quality, accuracy of angle measurement, and their ability to explain the trigonometric principles behind their gadget.

Homework

Assign a homework task that requires students to research and present a real-world application of trigonometry in various fields, such as engineering, physics, or architecture.

Conclusion

Conclude the lesson by summarizing the key trigonometric concepts learned and emphasizing their practical use in Arduino gadget development. Highlight the connection between mathematics and technology.

English

FROM GAME TO STEM

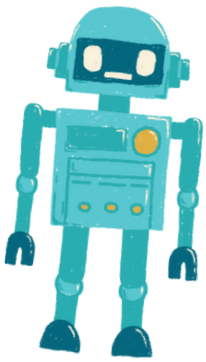
With the support of the Erasmus+ Programme of the European Union

Lesson 2
Arduino Math Lesson

Algebraic Modeling with Arduino

Duration:

2 periods



Objective:

- Students will learn how to use algebraic concepts to solve real-world problems.
- Students will apply mathematical modeling and programming skills to create an Arduino gadget that solves a practical problem.
- Students will gain experience with variables and equations.

Materials:

- Arduino boards (one per student or group)
- Breadboards
- Sensors or input devices (e.g., temperature sensor, light sensor, push button)
- LEDs, resistors, and jumper wires
- USB cables for programming
- Computers with Arduino IDE installed
- Projector or whiteboard for demonstrations

Activities

Day 1

Introduction to Algebraic Modeling (15 minutes):

- Start the lesson by introducing algebraic modeling and its relevance in solving real-world problems.
- Discuss the importance of variables and equations in mathematical modeling.

Variables and Equations (30 minutes):

- Review the concept of variables and how they are used to represent unknown values.
- Introduce linear equations (e.g., y=mx+b) as a way to model relationships between variables.
- Provide examples of real-world problems that can be represented using equations.

Arduino Basics (20 minutes):

- Introduce Arduino as a platform for creating gadgets and explain its components.
- Show examples of simple Arduino projects and how variables can be used in programming.

Hands-on Activity (45 minutes):

- Divide students into pairs or small groups.
- Provide each group with an Arduino board, a breadboard, a sensor or input device (e.g., temperature sensor), and LED.
- 1Instruct students to create a simple Arduino project that uses a sensor to read data and an LED to represent the data visually.
- Encourage students to use variables to store sensor readings and create equations to control the LED based on sensor data.
- Discuss different real-world scenarios where their gadget could be useful.

Day 2

Review of Algebraic Concepts (20 minutes):

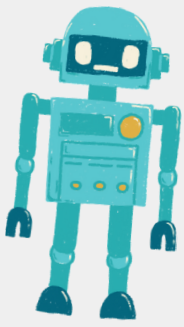
- Start the second day by reviewing the concepts of variables, equations, an mathematical modeling.
- Discuss the previous day's Arduino projects and their applications.

Programming the Algebraic Model (45 minutes):

- Continue with the hands-on activity from Day 1.
- Instruct students to modify their Arduino code to create a mathematical model using sensor data.
- Encourage them to experiment with different equations and variables to represent the relationship between sensor data and LED output.
- Discuss how the model can be used to make predictions or control real-world systems.

Here's an example of Arduino code for a simple algebraic modeling project. In this project, students will use a temperature sensor to read temperature data and control an LED based on the temperature reading. The code uses variables and an equation to determine when the LED should turn on or off.

```
// Define the pins for the temperature sensor, LED, and a resistor (if needed)
const int temperatureSensorPin = A0; // Analog pin for the temperature sensor
const int ledPin = 13; // Use the built-in LED on most Arduino boards
const float thresholdTemperature = 25.0; // Define the threshold temperature
void setup() {
pinMode(ledPin, OUTPUT); // Set the LED pin as an output
Serial.begin(9600); // Initialize serial communication for debugging (optional)
}
void loop() {
// Read the temperature from the sensor
int sensorValue = analogRead(temperatureSensorPin);
// Convert the analog value to temperature in degrees Celsius
float temperatureCelsius = map(sensorValue, 0, 1023, 0, 100); // Adjust the mapping as
needed
// Check if the temperature is above the threshold
if (temperatureCelsius > thresholdTemperature) {
digitalWrite(ledPin, HIGH); // Turn the LED on
} else {
digitalWrite(ledPin, LOW); // Turn the LED off
}
// Print the temperature to the serial monitor for debugging (optional)
Serial.print("Temperature: ");
Serial.print(temperatureCelsius);
Serial.println(" °C");
// Delay for a moment to avoid rapid LED toggling
delay(1000); // Delay for 1 second
}
```



In this code:

We define the pins for the temperature sensor (connected to an analog pin), the LED (usually the built-in LED on most Arduino boards), and a threshold temperature that determines when the LED should turn on.
In the setup() function, we set the LED pin as an output and initialize serial communication for debugging (optional).
In the loop() function, we read the temperature from the sensor using analogRead(). We convert the analog sensor value to temperature in degrees Celsius based on the sensor's characteristics.
We then check if the temperature exceeds the threshold defined (thresholdTemperature). If the temperature is above the threshold, the LED is turned on; otherwise, it's turned off.
Optional: We print the temperature to the Serial Monitor for debugging purposes.
To use this code, students will need to connect a temperature sensor (e.g., LM35) to an analog pin on the Arduino and an LED to a digital pin. The code will read the temperature from the sensor and control the LED based on the temperature reading and the threshold value.

Project Presentation and Discussion (30 minutes):

- Each group presents their Arduino gadget to the class.
- Encourage students to explain the mathematical model they created and how it works.
- Facilitate a class discussion on the applications of algebraic modeling in solving practical problems.

Assessments

Assess students based on their participation, the functionality of their Arduino gadget, and their ability to explain the algebraic concepts used in their project.

Homework

Assign a homework task that requires students to research and present a real-world problem that can be solved using algebraic modeling and an Arduino gadget.

Conclusion

Conclude the lesson by summarizing the key algebraic concepts learned and highlighting the importance of mathematical modeling in technology and engineering.

English

FROM GAME TO STEM

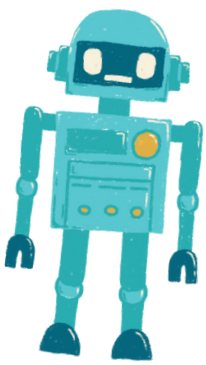
Co-funded by the
Erasmus+ Programme
of the European Union

Lesson 3
Arduino Physics Lesson

Exploring Motion and Acceleration with Arduino

Duration:

3 periods



Objective:

- Students will understand the basic principles of motion and acceleration.
- Students will apply kinematic equations to solve real-world problems related to motion.
- Students will program an Arduino to measure and display acceleration using a sensor.

Materials:

- Arduino boards (one per student or group)
- Accelerometer sensor modules (e.g., ADXL345)
- Breadboards
- Jumper wires
- USB cables for programming
- Computers with Arduino IDE installed
- Projector or whiteboard for demonstrations
- Optional: Real-life objects for motion experiments (e.g., toy cars, balls)

Activities

Day 1

Introduction to Motion and Acceleration (15 minutes):

- Begin the lesson by introducing the concept of motion and acceleration.
- Define key terms like velocity, acceleration, and deceleration.
- Discuss the importance of understanding motion in various fields, such as physics, engineering, and transportation.

Physics of Motion (30 minutes):

- Explain the equations of motion, including:

- Displacement: $s = ut + \frac{1}{2}at^2$
- Velocity: $v = u + at$

- Acceleration: $a = \frac{v-u}{t}$
- Provide exam ' these equations are used to analyze motion.
- Conduct simple motion experiments (e.g., rolling a ball down a slope) to demonstrate the principles of motion.

Introduction to Arduino and Accelerometers (20 minutes):

- Introduce Arduino as a platform for creating gadgets and explain its components.
- Explain the concept of accelerometers and how they measure acceleration.
- Show examples of accelerometer data output and explain how it relates to real-world motion.

Hands-on Activity (45 minutes):

- Divide students into pairs or small groups.
- Provide each group with an Arduino board, an accelerometer sensor module, a breadboard, and jumper wires.
- Instruct students to assemble a simple circuit to connect the accelerometer to the Arduino.
- Guide students in writing Arduino code to read and display acceleration data from the sensor in the Serial Monitor.

Day 2

Review of Kinematic Equations (20 minutes):

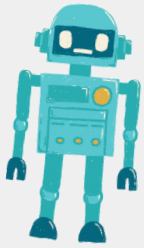
- Begin the second day by reviewing the kinematic equations discussed on Day 1.
- Provide additional examples for students to solve using these equations.

Programming the Acceleration Measurement Device (45 minutes):

- Continue with the hands-on activity from Day 1.
- Instruct students to complete their Arduino code to read and display real-time acceleration data from the accelerometer.
- Encourage students to calibrate the sensor if necessary and apply the acceleration equations to interpret the data.

Here's an example of Arduino code for measuring and displaying acceleration using an ADXL345 accelerometer sensor. This code will allow your students to interface the accelerometer sensor with the Arduino and display the acceleration values on the Serial Monitor.

```
#include <Wire.h>
#include <Adafruit_Sensor.h>
#include <Adafruit_ADXL345_U.h>
Adafruit_ADXL345_Unified accel = Adafruit_ADXL345_Unified(12345);
void setup(void) {
 Serial.begin(9600);
 Serial.println("Accelerometer Test"); Serial.println("");
  if(!accel.begin())
 {
/* There was a problem detecting the ADXL345 ... check your connections*/
   Serial.println("Ooops, no ADXL345 detected ... Check your wiring!");
   while(1);
 }
}
void loop(void) {
 sensors_event_t event;
 accel.getEvent(&event);
 /* Display the acceleration values (in m/s^2) */
 Serial.print("X: "); Serial.print(event.acceleration.x); Serial.print(" ");
 Serial.print("Y: "); Serial.print(event.acceleration.y); Serial.print(" ");
 Serial.print("Z: "); Serial.print(event.acceleration.z); Serial.println(" m/s^2");
 delay(500); // Delay for half a second between readings
}
```



In this code:
We include the necessary libraries for the ADXL345 accelerometer sensor.
We create an Adafruit_ADXL345_Unified object called accel to interface with the sensor.
In the setup() function, we initialize the serial communication for debugging and check if the accelerometer sensor is detected. If not, it will display an error message.
In the loop() function, we continuously read the acceleration data from the sensor using accel.getEvent(&event) and display the X, Y, and Z-axis acceleration values in meters per second squared (m/s^2) on the Serial Monitor.
To use this code, make sure your students have connected the ADXL345 accelerometer sensor to the Arduino correctly. The sensor should be connected to the appropriate pins (e.g., SDA and SCL) for I2C communication. When the Arduino is powered on and running this code, it will continuously display the acceleration data on the Serial Monitor.
Please note that you may need to install the Adafruit ADXL345 library via the Arduino Library Manager to use this code.

Project Presentation and Discussion (60 minutes):

- Each group presents their Arduino-based acceleration measurement device to the class.
- Encourage students to explain how they applied the kinematic equations and physics principles in their projects.
- Discuss the real-world applications of acceleration measurement devices in various fields.
- Facilitate a class discussion on the challenges and solutions encountered during the project.

Assessments

Assess students based on their participation, code quality, accuracy of acceleration measurement, and their ability to explain the physics principles behind their gadget.

Homework

Assign a homework task that requires students to research and present a real-world application of acceleration measurement in physics or engineering

Conclusion

Conclude the lesson by summarizing the key physics concepts learned and emphasizing their practical use in Arduino gadget development. Highlight the connection between physics and technology.

English

FROM GAME TO STEM

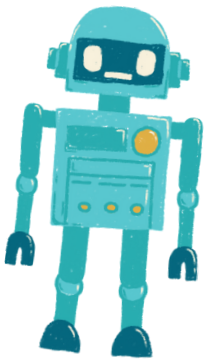
Co-funded by the
Erasmus+ Programme
of the European Union

Lesson 4
Arduino Physics Lesson

Exploring Oscillations and Pendulum Motion with Arduino

Duration:

2 periods



Objective:

- Students will understand the principles of oscillatory motion and harmonic oscillations.
- Students will apply mathematical modeling to analyze pendulum motion.
- Students will program an Arduino to simulate and visualize pendulum motion.

Materials:

- Arduino boards (one per student or group)
- Servo motors
- Breadboards
- Jumper wires
- Small objects as pendulum weights (e.g., washers)
- String or thread for pendulums
- Rulers or measuring tape
- USB cables for programming
- Computers with Arduino IDE installed
- Projector or whiteboard for demonstrations

Activities

Day 1

Introduction to Oscillations and Pendulum Motion (15 minutes):

- Begin the lesson by introducing the concept of oscillatory motion and its relevance in various fields, including physics and engineering.
- Define key terms like period, frequency, amplitude, and harmonic oscillations.
- Explain the basics of pendulum motion and its mathematical representation.

Mathematical Modeling of Pendulums (30 minutes):

- Discuss the mathematical model of a simple pendulum, including the equation for the period of a pendulum: $T = 2\pi\sqrt{\frac{L}{g}}$

where T is the period, L is the length of the pendulum, and g is the acceleration due to gravity.

- Provide examples of how to use the equation to calculate the period of a pendulum.
- Conduct simple calculations related to pendulum motion.

Introduction to Arduino and Servo Motors (20 minutes):

- Introduce Arduino as a platform for creating gadgets and explain its components (microcontroller, sensors, actuators).
- Explain the concept of servo motors and how they can be used to simulate pendulum motion.
- Show examples of servo motor control with Arduino.

Hands-on Activity (45 minutes):

- Divide students into pairs or small groups.
- Provide each group with an Arduino board, a servo motor, a breadboard, jumper wires, a small object as a pendulum weight, and a string.
- Instruct students to assemble a simple pendulum simulator using the servo motor to control the motion of the pendulum.
- Guide students in writing Arduino code to create harmonic oscillations by varying the servo's position.

Day 2

Review of Pendulum Motion (20 minutes):

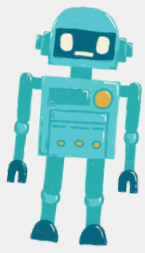
- Begin the second day by reviewing the concepts of oscillatory motion, pendulum motion, and the mathematical model of a simple pendulum.
- Discuss the previous day's Arduino projects and their applications.

Programming the Pendulum Simulator (45 minutes):

- Continue with the hands-on activity from Day 1.
- Instruct students to modify their Arduino code to accurately simulate pendulum motion with various parameters such as length and amplitude.
- Encourage students to visualize and graphically represent the motion using the servo motor.

Here's an example of Arduino code for creating a simple pendulum simulator using a servo motor. This code allows students to program an Arduino to simulate and visualize the motion of a pendulum:

```
#include <Servo.h>
Servo pendulum; // Create a Servo object
int angle = 90; // Initial angle of the pendulum (straight down)
int amplitude = 45; // Maximum angle to one side of the vertical (adjust as needed)
int period = 2000; // Period of the pendulum swing in milliseconds (adjust as needed)
void setup() {
  pendulum.attach(9); // Attach the servo to digital pin 9
}
void loop() {
  // Calculate the angle of the pendulum using harmonic motion
  int displacement = amplitude * cos(2 * PI * millis() / period);
  int pendulumAngle = angle + displacement;
   // Move the servo to the calculated angle
  pendulum.write(pendulumAngle);
  // Delay for a short time to control the speed of the simulation
  delay(50); // Adjust as needed for desired speed
}
```



In this code:

- We include the Servo library to control the servo motor.
- We create a Servo object called pendulum to control the servo motor.
- We define variables for the initial angle of the pendulum (angle), the maximum angle to one side of the vertical (amplitude), and the period of the pendulum swing (period). You can adjust these values to change the behavior of the pendulum.
- In the setup() function, we attach the servo to digital pin 9.
- In the loop() function, we calculate the angle of the pendulum using the formula for harmonic motion. The displacement represents the angular displacement from the vertical position, and we add it to the initial angle to get the pendulumAngle.
- We use pendulum.write(pendulumAngle) to move the servo to the calculated angle.
- We add a delay to control the speed of the simulation. Adjust the delay value as needed to achieve the desired simulation speed.

To use this code, students should connect a servo motor to digital pin 9 on the Arduino and upload the code to the board. The servo motor will simulate the motion of a pendulum, and students can observe the oscillations in action.

Project Presentation and Discussion (30 minutes):

- Each group presents their Arduino-based pendulum simulator to the class.
- Encourage students to explain how they applied mathematical modeling principles in their projects.
- Discuss the real-world applications of understanding harmonic oscillations in fields like physics, engineering, and astronomy.
- Facilitate a class discussion on the challenges and solutions encountered during the project.

Assessments

Assess students based on their participation, the quality of their Arduino gadget, and their ability to explain the principles of oscillatory motion and harmonic oscillations.

Homework

Assign a homework task that requires students to research and present a real-world application of oscillations and harmonic motion in science or technology.

Conclusion

Conclude the lesson by summarizing the key physics concepts learned and emphasizing their practical use in Arduino gadget development. Highlight the connection between mathematics and technology in understanding pendulum motion.

English

FROM GAME TO STEM

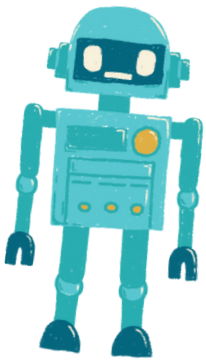
Co-funded by the Erasmus+ Programme of the European Union

Lesson 5
Arduino Chemistry Lesson

Exploring Hydroponics and Plant Growth Chemistry

Duration:

3 periods



Objective:

- Students will understand the principles of hydroponics and its advantages in plant growth.
- Students will learn about essential nutrients for plant growth and how to prepare nutrient solutions.
- Students will design and build an automated irrigation system using Arduino.
- Students will monitor and control pH levels in a hydroponic system to optimize plant growth.

Materials:

- Arduino boards (one per student or group)
- pH sensors and pH calibration solutions
- Water pumps
- Tubing and drip emitters
- Reservoir for nutrient solution
- pH up and pH down solutions
- pH buffer solutions
- Plant seeds or seedlings
- Hydroponic growing medium (e.g., rockwool, perlite)
- Nutrient solution components (e.g., N-P-K fertilizer)
- Containers for hydroponic setups
- Jumper wires
- USB cables for programming
- Computers with Arduino IDE installed
- Projector or whiteboard for demonstrations

Activities

Day 1

Introduction to Hydroponics (15 minutes):

- Begin the lesson by introducing the concept of hydroponics and its advantages in plant growth.
- Discuss the benefits of controlled environments and water-efficient systems.

Chemistry of Plant Growth (30 minutes):

- Explain the chemical elements essential for plant growth (e.g., nitrogen, phosphorus, potassium) and their roles.
- Discuss the importance of pH levels in nutrient uptake and plant health.
- Introduce the concept of nutrient solutions and pH control in hydroponics.

Introduction to Arduino and Sensors (20 minutes):

- Introduce Arduino as a platform for automation and data collection and explain its components (microcontroller, sensors).
- Show examples of sensors used in hydroponic systems (e.g., pH sensors).

Hands-on Activity Preparation (45 minutes):

- Provide students with Arduino boards, pH sensors, water pumps, tubing, and containers.
- Instruct students to brainstorm and plan their hydroponic system design, including nutrient solutions and pH control.

Day 2

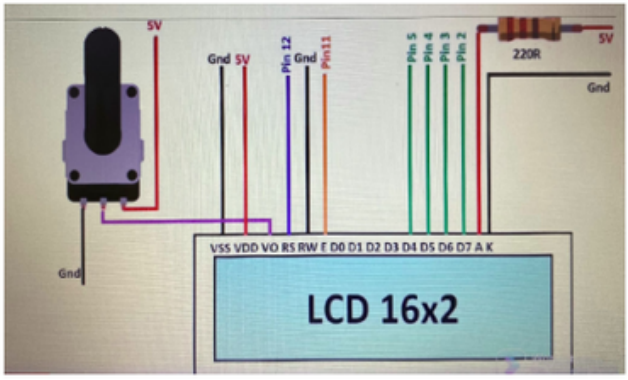
Hydroponic System Setup (60 minutes):

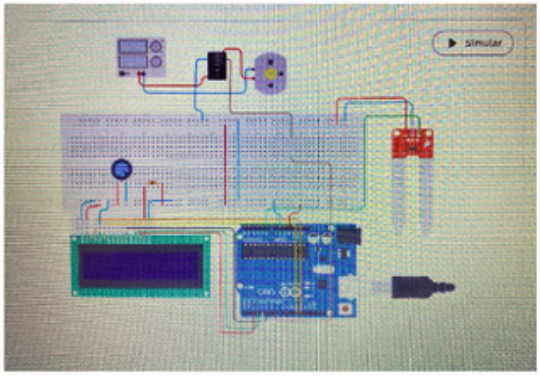
- Begin the second day by allowing students to set up their hydroponic systems.
- Students should plant seeds or seedlings in a hydroponic growing medium (e.g., rockwool) and arrange the irrigation system with tubing and drip emitters.
- Demonstrate how to mix and prepare nutrient solutions.

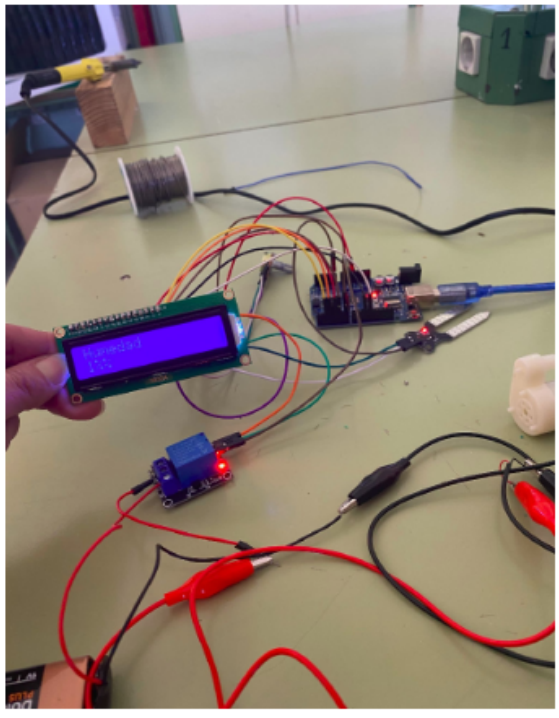
pH Monitoring and Calibration (30 minutes):

- Instruct students on how to calibrate and use pH sensors for pH monitoring.
- Explain the importance of maintaining the pH within the optimal range for nutrient uptake (typically around pH 6-7).
- Guide students in calibrating their pH sensors using pH buffer solutions.

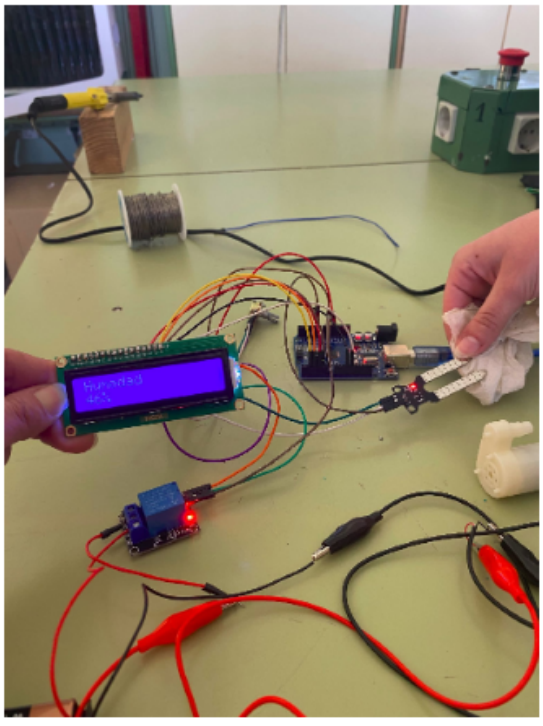
Sheme of the conections:







1% Humidity: The pump works



46% Humidity: The pump keeps working

Day 3

Arduino Programming (60 minutes):

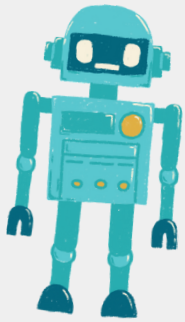
- Instruct students to write Arduino code for the automated irrigation system.
- Students should program the Arduino to control the water pump's schedule and monitor pH levels.
- Emphasize the importance of regular watering and pH adjustments for plant growth.

Data Collection and pH Control (30 minutes):

- Explain how to collect data from the pH sensors and monitor pH levels.
- Discuss the actions students should take if pH levels deviate from the optimal range.

Here's an example of Arduino code for an automated hydroponic irrigation system with pH monitoring and control. This code is a basic framework that you can adapt and expand upon based on your specific setup and needs. It includes pH monitoring, pH adjustment, and controlling a water pump for irrigation.

```
#include <Adafruit_ADS1015.h>
#include <Wire.h>
#define PH_SENSOR_PIN 0 // Analog pin for pH sensor
#define PUMP_PIN 12    // Digital pin for water pump
// Create an Adafruit ADS1015 ADC object
Adafruit_ADS1015 ads;
float currentpH;
float targetpH = 6.5; // Adjust this value to your desired pH level
void setup() {
 Serial.begin(9600);
 ads.begin();
 pinMode(PUMP_PIN, OUTPUT);
 digitalWrite(PUMP_PIN, LOW); // Initialize pump as off
}
void loop() {
 // Read pH value from pH sensor
 currentpH = readpH();
 // Display current pH value
 Serial.print("Current pH: ");
 Serial.println(currentpH);
 // Check and adjust pH level
 if (currentpH < targetpH) {
   // pH is too low, add pH up solution (adjust as needed)
   // Implement pH adjustment mechanism here
     // For example, you can control a peristaltic pump for adding pH up solution
```



```
digitalWrite(PUMP_PIN, HIGH); // Turn on the pump
 delay(1000); // Adjust the delay time as needed
 digitalWrite(PUMP_PIN, LOW); // Turn off the pump
 } else if (currentpH > targetpH) {
 // pH is too high, add pH down solution (adjust as needed)
 // Implement pH adjustment mechanism here
  // For example, you can control a peristaltic pump for adding pH down solution
 digitalWrite(PUMP_PIN, HIGH); // Turn on the pump
 delay(1000); // Adjust the delay time as needed
 digitalWrite(PUMP_PIN, LOW); // Turn off the pump
 }
 // Add code for irrigation control here (e.g., based on time intervals)
}
float readpH() {
 // Read pH value from the pH sensor and convert to pH scale
 int16_t rawValue = ads.readADC_SingleEnded(PH_SENSOR_PIN);
 float voltage = (rawValue * 0.1875) / 1000.0; // Convert to voltage
  float pHValue = 3.5 * voltage + 3.5; // Convert to pH scale (adjust calibration values as needed)
 return pHValue;
}
```

In this code:

We use the Adafruit ADS1015 library to interface with the ADS1015 ADC, which is used to read pH sensor values. Make sure you have this library installed in your Arduino IDE.

The setup() function initializes serial communication for data output, initializes the ADC, and configures the water pump pin as an output.

In the loop() function, we continuously read the pH value from the pH sensor using the readpH() function.

We compare the current pH value to the target pH level (targetpH) and adjust the pH as needed. You should implement the pH adjustment mechanism based on your specific setup, which may involve controlling a peristaltic pump for adding pH up or pH down solutions.

Additionally, you can add code for controlling the water pump for irrigation based on time intervals or other criteria.

Please note that this code provides a basic framework, and you may need to calibrate and fine-tune it according to your specific sensor and pump setup, as well as the desired pH levels and irrigation schedule. Ensure safety precautions are taken when working with chemicals and pumps.

Day 4

Experiment Setup and Monitoring (60 minutes):

- Allow students to set up their automated hydroponic systems in the greenhouse or classroom.
- Students should start the system and monitor plant growth, nutrient levels, and pH.

Project Presentation and Discussion (60 minutes):

- Each group presents their hydroponic system design, methodology, and initial results to the class.
- Discuss the impact of nutrient solutions and pH control on plant growth.
- Encourage students to propose optimizations for their systems based on initial observations.

Assessments

Assess students based on their participation, system setup, pH calibration, data collection, analysis, and the quality of their presentations.

Homework

Assign homework that requires students to research and present a real-world application of hydroponics in agriculture or sustainable farming.

Conclusion

Conclude the lesson by summarizing the key chemistry concepts learned and reinforcing the significance of hydroponics in sustainable agriculture and food production. Highlight the role of technology (Arduino) in automating and optimizing hydroponic systems.

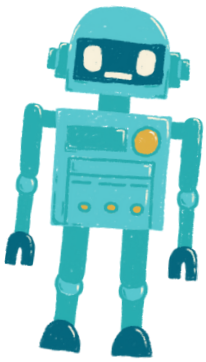
English

Lesson 6
Arduino Chemistry Lesson

Exploring Chemical Reactions with Arduino

Duration:

3 periods



Objective:

- Students will understand the principles of chemical reactions and their relevance in daily life.
- Students will design and conduct experiments to observe and measure chemical reactions.
- Students will program an Arduino-based temperature monitoring system to collect and analyze data from chemical reactions.

Materials:

- Arduino boards (one per student or group)
- Temperature sensors (e.g., DS18B20 or LM35)
- Chemicals for experiments (e.g., baking soda, vinegar)
- Reaction containers (e.g., beakers, test tubes)
- Jumper wires
- USB cables for programming
- Computers with Arduino IDE installed
- Projector or whiteboard for demonstrations
- Safety goggles and lab coats

Activities

Day 1

Introduction to Chemical Reactions (15 minutes):

- Begin the lesson by introducing the concept of chemical reactions and their role in everyday life.
- Discuss the importance of chemical reactions in various fields, including chemistry, biology, and industry.

Chemical Reaction Basics (30 minutes):

- Dive into the fundamentals of chemical reactions, including reactants, products, and the conservation of mass.
- Provide examples of common chemical reactions and their applications.
- Emphasize safety protocols when handling chemicals in experiments.

Introduction to Arduino and Sensors (20 minutes):

- Introduce Arduino as a platform for data collection and explain its components (microcontroller, sensors).
- Show examples of temperature sensors and how they can be connected to Arduino for data collection.

Hands-on Activity Preparation (45 minutes):

- Provide students with Arduino boards, temperature sensors, chemicals, and reaction containers.
- Instruct students to brainstorm and plan their chemical reaction experiments, focusing on temperature changes.
- Discuss safety precautions and lab rules.

Day 2

Experiment Setup and Data Collection (60 minutes):

- Begin the second day by allowing students to set up their chemical reactio experiments.
- Students should mix the selected chemicals in reaction containers and position the temperature sensors.
- Instruct students to write Arduino code for temperature monitoring and data collection.

Data Analysis and Discussion (30 minutes):

- Guide students in analyzing their data, looking for temperature changes during the chemical reactions.
- Discuss the significance of temperature changes in chemical reactions and the concepts of exothermic and endothermic reactions.
- Encourage students to draw conclusions based on their findings.

Day 3

Programming the Monitoring System (60 minutes):

- Instruct students to fine-tune their Arduino code for data collection and analysis.
- Students should program the Arduino to record temperature data at specific intervals and display it graphically (e.g., on an LCD screen or the Serial Monitor).

Project Presentation and Discussion (60 minutes):

- Each group presents their chemical reaction experiment setup, methodology, and results to the class.
- Encourage students to explain their observations and the implications of temperature changes in chemical reactions.
- Facilitate a class discussion on the real-world applications of chemical reactions in various fields.

Here's an example of Arduino code that you can use for temperature monitoring during a chemical reaction experiment. This code reads temperature data from a DS18B20 temperature sensor and displays it on the Serial Monitor. You can customize and expand upon this code to suit your specific experiment.

```
#include <OneWire.h>
#include <DallasTemperature.h>
// Data wire is plugged into digital pin 2
#define ONE_WIRE_BUS 2
// Create a OneWire object
OneWire oneWire(ONE_WIRE_BUS);
// Pass the OneWire reference to Dallas Temperature sensor
DallasTemperature sensors(&oneWire);
```

```
void setup() {
 // Initialize serial communication for data output
 Serial.begin(9600);

 // Start the temperature sensor library
 sensors.begin();
}

void loop() {
 // Request temperature readings
 sensors.requestTemperatures();

 // Read temperature in Celsius
 float temperatureC = sensors.getTempCByIndex(0);

 // Display temperature on the Serial Monitor
 Serial.print("Temperature: ");
 Serial.print(temperatureC);
 Serial.println(" °C");

 // Add code here for data storage or further analysis

 // Delay before the next temperature reading (adjust as needed)
 delay(1000); // 1-second delay
}
```

In this code:

We use the Dallas Temperature library to interface with the DS18B20 temperature sensor. Make sure you have this library installed in your Arduino IDE.

The setup() function initializes serial communication for data output and starts the temperature sensor library.

In the loop() function, the program continuously requests and reads temperature data from the sensor using sensors.getTempCByIndex(0). The 0 indicates the first (and in this case, only) temperature sensor connected.

The temperature data is displayed on the Serial Monitor with a delay of 1 second between readings. You can adjust the delay time to control the data collection frequency.

You can modify this code to include additional sensors for different experiments, implement data storage to an SD card, or create graphical representations of the data on an LCD screen, depending on your specific requirements.

Assessments

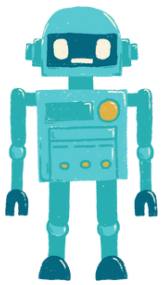
Assess students based on their participation in the experiment, data collection, analysis, and the quality of their presentations.

Homework

Assign homework that requires students to research and present a real-world application of chemical reactions in a specific industry or field of science.

Conclusion

Conclude the lesson by summarizing the key chemistry concepts learned and reinforcing the importance of chemical reactions in understanding natural processes and technological advancements. Highlight the role of technology (Arduino) in scientific investigations.



English

FROM GAME TO STEM

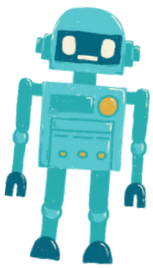
Co-funded by the
Erasmus+ Programme
of the European Union

Lesson 7
Arduino Biology Lesson

Investigating Thermal Conductivity with Arduino Thermometers

Duration:

3 periods



Objective:

- Students will understand the concept of thermal conductivity and its importance.
- Students will design and conduct an experiment to determine the thermal conductivity of different materials.
- Students will utilize Arduino thermometers to collect temperature data and analyze their findings.

Materials:

- Arduino boards (one per student or group)
- Temperature sensors (e.g., DS18B20 or LM35)
- Breadboards
- Jumper wires
- Various materials for testing conductivity (e.g., metals, plastics, wood, glass)
- Insulating materials (e.g., foam, cloth)
- Hot water or a heat source
- Cold water or ice
- Stopwatch or timer
- USB cables for programming
- Computers with Arduino IDE installed
- Projector or whiteboard for demonstrations
- Safety goggles and gloves (for handling hot materials)

Activities

Day 1

Introduction to Thermal Conductivity (15 minutes):

- Begin the lesson by introducing the concept of thermal conductivity and why it is important in everyday life.
- Discuss real-world applications of thermal conductivity, such as in cooking, building materials, and engineering.

Heat Transfer and Insulation (30 minutes):

- Explain the different methods of heat transfer (conduction, convection, radiation) and focus on conduction, which is relevant to the experiment.
- Discuss insulating materials and their role in reducing heat transfer.
- Emphasize the need for a controlled experiment to measure thermal conductivity.

Introduction to Arduino Thermometers(20 minutes):

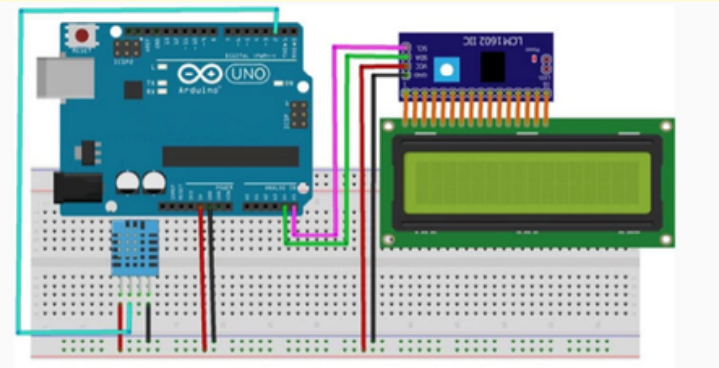
- Introduce Arduino as a platform for data collection and explain its components (microcontroller, sensors).
- Explain the use of temperature sensors and how they can be connected to Arduino boards.
- Show examples of temperature data collection and visualization with Arduino.

Hands-on Activity Preparation(45 minutes):

- Provide students with Arduino boards, temperature sensors, breadboards, and jumper wires.
- Explain the experimental setup: Each group will test different materials to determine their thermal conductivity.
- Instruct students to brainstorm and plan their experiments, including how to control variables and collect data.

English

SHEMA:



Day 2

Experiment Setup and Data Collection (60 minutes):

- Begin the second day by allowing students to set up their experiments.
- Students should attach the temperature sensor to their selected materials and prepare containers of hot and cold water.
- Instruct students to start collecting temperature data using Arduino and record the initial temperatures.
- Have students measure and record the time it takes for the temperature to change in each material.

Data Analysis and Discussion(30 minutes):

- Guide students in analyzing their data, calculating the rate of temperature change, and comparing the conductivity of different materials.
- Discuss the concept of thermal resistance and how it relates to conductivity.
- Encourage students to identify which material is the best heat conductor and which is the worst based on their findings.

Here's an example of Arduino code that you can use for the experiment to measure and compare the thermal conductivity of different materials using a temperature sensor (e.g., DS18B20). This code will collect temperature data from the sensor and allow you to calculate the rate of temperature change for each material.

```
#include <OneWire.h>
#include <DallasTemperature.h>
// Data wire is plugged into digital pin 2
#define ONE_WIRE_BUS 2
// Create a OneWire object
OneWire oneWire(ONE_WIRE_BUS);
// Pass the OneWire reference to Dallas Temperature sensor
DallasTemperature sensors(&oneWire);
void setup() {
 // Initialize serial communication for data output
 Serial.begin(9600);
 // Start the temperature sensor library
 sensors.begin();
}
void loop() {
```

```
// Initialize variables for temperature measurements
 float initialTemp, finalTemp;
 unsigned long startTime, endTime;
 float rateOfChange;
 // Wait for the user to start the experiment (e.g., press a button)
while (!digitalRead(3)) {
  // Wait for button press
 }
 delay(1000); // Debounce delay
 // Measure initial temperature
 sensors.requestTemperatures(); // Request temperature readings
 initialTemp = sensors.getTempCByIndex(0); // Get temperature in Celsius
 // Record the start time
 startTime = millis();
 // Wait for the temperature to stabilize (e.g., 5 seconds)
 delay(5000);
 // Measure final temperature
 sensors.requestTemperatures();
 finalTemp = sensors.getTempCByIndex(0);
 // Record the end time
 endTime = millis();
 // Calculate the rate of temperature change (°C per second)
  rateOfChange = (finalTemp - initialTemp) / ((endTime - startTime) /
1000.0);
 // Output results to the Serial Monitor
 Serial.print("Initial Temperature: ");
 Serial.print(initialTemp);
 Serial.println(" °C");
 Serial.print("Final Temperature: ");
 Serial.print(finalTemp);
 Serial.println(" °C");
 Serial.print("Rate of Change: ");
 Serial.print(rateOfChange);
 Serial.println(" °C/s");
 // Wait for user input (e.g., press a button) to proceed to the next material
 Serial.println("Press a button to test the next material.");
 while (!digitalRead(3)) {
  // Wait for button press
 }
 delay(1000); // Debounce delay
}
```

In this code:
We use the Dallas Temperature library to interface with the DS18B20 temperature sensor. Make sure you have this library installed in your Arduino IDE.
We define the digital pin (e.g., pin 2) to which the data wire of the DS18B20 sensor is connected.
In the setup() function, we initialize serial communication for data output and start the temperature sensor library.
In the loop() function, the program waits for a button press to start the experiment. The temperature is measured before and after a waiting period (e.g., 5 seconds) to calculate the rate of temperature change.
The rate of change is calculated as the difference in temperature divided by the time taken to reach that change. This gives you the rate in °C per second.
Results are printed to the Serial Monitor, including the initial temperature, final temperature, and rate of change.
After the experiment for one material is completed, you can press a button to proceed to the next material.
Remember to connect the DS18B20 sensor to the Arduino correctly, and ensure that the button is connected to a digital pin (e.g., pin 3) for triggering the experiment. Adjust the code as needed based on your specific setup.

Day 3

Project Presentation and Discussion (60 minutes):

- Each group presents their experiment setup, methodology, and results to the class.
- Encourage students to explain their observations, discuss potential sources of error, and suggest improvements.
- Facilitate a class discussion on the real-world implications of their findings, such as selecting materials for thermal insulation or conducting materials for heat sinks.

Assessments

Assess students based on their participation in the experiment, data collection, analysis, and the quality of their presentations.

Homework

Assign homework that requires students to research and present a real-world application of thermal conductivity in engineering or materials science.

Conclusion

Conclude the lesson by summarizing the key concepts learned and reinforcing the importance of understanding thermal conductivity in everyday life and technological applications.

English

FROM GAME TO STEM

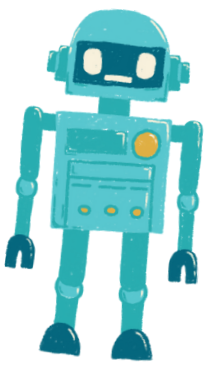
Co-funded by the
Erasmus+ Programme
of the European Union

Lesson 8
Arduino Biology Lesson

Exploring Photosynthesis and Plant Growth with Arduino

Duration:

3 periods



Objective:

- Students will understand the process of photosynthesis and its significance in plant growth.
- Students will design and conduct an experiment to investigate the effect of environmental factors on photosynthesis.
- Students will program an Arduino-based system to monitor and collect data related to plant growth.

Materials:

- Arduino boards (one per student or group)
- Sensors (e.g., light sensor, temperature sensor, humidity sensor)
- LED grow lights (optional)
- Potted plants or seeds
- Soil and planting containers
- Jumper wires
- USB cables for programming
- Computers with Arduino IDE installed
- Projector or whiteboard fo demonstrations
- Potting soil, water, and other materials for plant care

Activities

Day 1

Introduction to Photosynthesis (15 minutes):

- Begin the lesson by introducing the concept of photosynthesis and its importance in plant growth.
- Discuss the chemical equation of photosynthesis and the role of light, carbon dioxide, and water in the process.

Environmental Factors and Plant Growth (30 minutes):

- Explain how various environmental factors, such as light intensity, temperature, and humidity, can affect photosynthesis and plant growth.
- Discuss why these factors are essential for healthy plant development.
- Emphasize the need for controlled experiments to study these factors.

Introduction to Arduino and Sensors (20 minutes):

- Introduce Arduino as a platform for data collection and explain its components (microcontroller, sensors).
- Show examples of sensors commonly used in environmental monitoring and plant care.
- Explain the role of sensors in collecting data for experiments.

Hands-on Activity Preparation(45 minutes):

- Provide students with Arduino boards, sensors (e.g., light sensor, temperature sensor, humidity sensor), and LED grow lights (optional).
- Instruct students to brainstorm and plan their plant growth experiments, focusing on one environmental factor.
- Have students prepare pots with soil and plant seeds or small potted plants.

Day 2

Experiment Setup and Data Collection (60 minutes):

- Begin the second day by allowing students to set up their experiments.
- Students should place the sensors in the plant environment, connect them to Arduino, and position the LED grow lights (if using).
- Instruct students to collect data related to the chosen environmental factor, such as light intensity or temperature.
- Emphasize the importance of recording data accurately and regularly.

Data Analysis and Discussion(30 minutes):

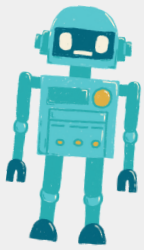
- Guide students in analyzing their data, looking for trends or patterns related to plant growth and the chosen factor.
- Discuss the impact of the factor on photosynthesis and plant development.
- Encourage students to draw conclusions from their findings.

Day 3

Programming the Monitoring System (60 minutes):

- Instruct students to write Arduino code to monitor and collect data from the sensors.
- Students should program the Arduino to record data at specific intervals (e.g., every hour).
- Discuss how to use libraries for sensor data retrieval and how to store data.

Here's an example of Arduino code for a simple environmental monitoring system that measures and logs data related to light intensity using a light sensor. You can adapt this code for other environmental factors as needed:



```
#include <Wire.h>
#include <Adafruit_Sensor.h>
#include <Adafruit_TSL2591.h>
Adafruit_TSL2591 tsl = Adafruit_TSL2591(2591);
void setup() {
 Serial.begin(9600);
  // Initialize the light sensor
 if(!tsl.begin()) {
   Serial.println("Light sensor not found. Check wiring.");
   while(1);
 }
 tsl.setGain(TSL2591_GAIN_LOW);    // Adjust the gain (options: LOW, MED, HIGH, MAX)
 tsl.setTiming(TSL2591_INTEGRATIONTIME_100MS); // Adjust the integration time (options:
100, 200, 300, 400, 500, 600)
}
void loop() {
 // Read and print light intensity data
 uint16_t luminance = tsl.getLuminosity(TSL2591_VISIBLE);
 Serial.print("Light Intensity (Lux): ");
 Serial.println(luminance);
    // Add code here to log data to an SD card, display on an LCD, or transmit to a
computer/server.
 // Wait for a specific time interval (e.g., 1 hour)
 delay(3600000); // Adjust the delay time as needed
}
```

In this code:

We use the Adafruit TSL2591 library to interface with the TSL2591 light sensor. Make sure you have this library installed in your Arduino IDE.
The setup() function initializes the serial communication for data output and sets up the light sensor. It also configures the gain and integration time of the sensor based on your requirements.
In the loop() function, the program continuously reads the light intensity data (in lux) from the sensor using tsl.getLuminosity(TSL2591_VISIBLE).
You can add code within the loop to log the data to an SD card, display it on an LCD screen, or transmit it to a computer or server for further analysis. For example, you can use an SD card module to store data locally.
The delay at the end of the loop is set to wait for a specific time interval (e.g., 1 hour) before taking the next reading. You can adjust the delay time to control the data collection frequency.

This code provides a basic framework for monitoring light intensity, and you can extend it by adding more sensors to monitor additional environmental factors like temperature and humidity. Remember to adjust the code to match the specific sensors and hardware you're using in your experiment.

Project Presentation and Discussion (60 minutes):

- Each group presents their plant growth experiment setup, methodology, and results to the class.
- Encourage students to explain their observations and the implications of their findings for plant care and agriculture.
- Facilitate a class discussion on the significance of photosynthesis in the ecosystem and how technology (Arduino) can aid in scientific investigations.

Assessments

Assess students based on their participation in the experiment, data collection, analysis, and the quality of their presentations.

Homework

Assign homework that requires students to research and present a real-world application of photosynthesis and environmental monitoring in agriculture or ecology.

Conclusion

Conclude the lesson by summarizing the key biology concepts learned and reinforcing the importance of photosynthesis and environmental factors in plant growth. Highlight the role of technology in advancing scientific understanding.

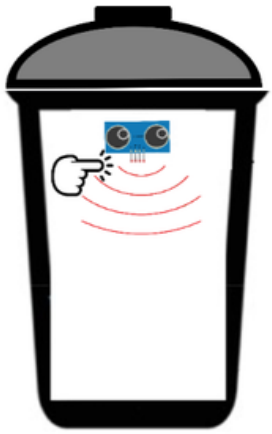
English

FROM GAME TO STEM

Co-funded by the
Erasmus+ Programme
of the European Union

Lesson 9
Arduino Ecology Lesson

Building a Smart Dustbin with Arduino



Duration:

3 periods

Objective:

- Students will understand the basics of Arduino programming and its applications in automation.
- Students will learn about Ultrasonic sensors and how they can be used for object detection.
- Students will build a Smart Dustbin prototype and program it to open its flap when an object is detected.
- Students will explore the environmental benefits of smart waste management systems.

Materials:

- Arduino Uno boards (one per student or group)
- Ultrasonic sensor HC-SR04 (one per student or group)
- Servomotors (one per student or group)
- Breadboards and jumper wires
- Small cardboard box or container (for the dustbin)
- Cardboard or plastic flap (to simulate the dustbin's flap)
- USB cables for programming
- Computers with Arduino IDE installed
- Projector or whiteboard for demonstrations

Activities

Day 1

Introduction to Arduino and Electronic (15 minutes):

- Begin the lesson by introducing Arduino as a microcontroller platform used for various projects.
- Discuss the significance of electronics and automation in modern technology.

Ultrasonic Sensors and Object Detection (30 minutes):

- Explain the principle of Ultrasonic sensors and how they work for distance measurement.
- Discuss the HC-SR04 Ultrasonic sensor's components (transmitter and receiver).
- Illustrate how Ultrasonic sensors can be used to detect objects and measure distances.

Introduction to Servomotors (20 minutes):

- Introduce Servomotors and their applications in controlling mechanical movements.
- Show examples of how Servomotors can be used in projects such as opening a flap.

Hands-on Activity Setup (45 minutes):

- Provide each group with an Arduino Uno, an Ultrasonic sensor, a Servomotor, a breadboard, and jumper wires.
- Instruct students to assemble the Smart Dustbin prototype, positioning the Ultrasonic sensor and Servomotor appropriately.

Day 2

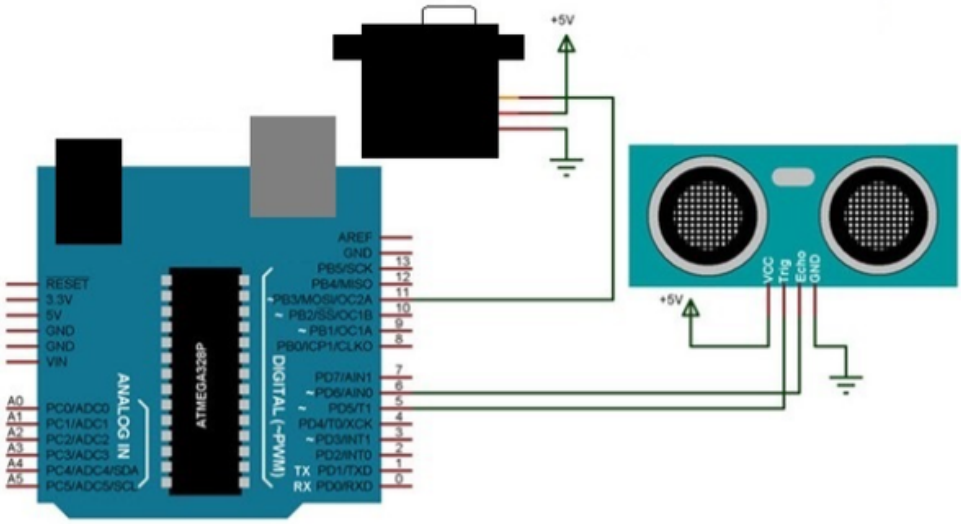
Arduino Programming Basics (30 minutes):

- Teach students the fundamentals of Arduino programming, including setup(), loop(), and pinMode().
- Provide examples of basic Arduino code for LED blinking.

Programming the Smart Dustbin (60 minutes):

- Guide students in writing Arduino code to control the Smart Dustbin based on Ultrasonic sensor readings.
- Explain the logic for opening the flap when an object is detected within a specified range.

SHEMA:



Here's an example Arduino code for a Smart Dustbin using an Ultrasonic sensor (HC-SR04) and a Servomotor. This code allows the Servomotor to open the dustbin's flap when an object is detected within a specified range:

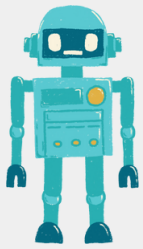
```
#include <Servo.h>

#define TRIGGER_PIN 9
#define ECHO_PIN 10
#define SERVO_PIN 11

Servo myservo; // Create a Servo object
int distance; // Variable to store distance measured by the Ultrasonic sensor

void setup() {
 myservo.attach(SERVO_PIN); // Attach the Servo to the specified pin
 pinMode(TRIGGER_PIN, OUTPUT);
 pinMode(ECHO_PIN, INPUT);
 Serial.begin(9600); // Initialize serial communication for debugging
}
```

```
void loop() {
 // Send a brief pulse to trigger the Ultrasonic sensor
 digitalWrite(TRIGGER_PIN, LOW);
 delayMicroseconds(2);
 digitalWrite(TRIGGER_PIN, HIGH);
 delayMicroseconds(10);
 digitalWrite(TRIGGER_PIN, LOW);
 // Read the duration of the echo pulse and calculate the distance
 duration = pulseIn(ECHO_PIN, HIGH);
 distance = (duration / 2) / 29.1; // Calculate distance in centimeters
 // Check if an object is within the specified range (adjust as needed)
 if (distance < 20) { // You can adjust the distance threshold here
  // If an object is detected, open the flap
   myservo.write(90); // Rotate the Servo to open the flap (adjust the angle as needed)
  delay(1000); // Wait for 1 second
  myservo.write(0); // Rotate the Servo back to close the flap
 }
 // Print the distance to the Serial Monitor for debugging
 Serial.print("Distance: ");
 Serial.print(distance);
 Serial.println(" cm");
 // Add a delay between readings to prevent rapid triggering
 delay(1000); // You can adjust the delay time as needed
}
```



In this code:
We include the Servo library and define the pin numbers for the Ultrasonic sensor's trigger and echo pins, as well as the Servomotor's control pin.
In the setup() function, we attach the Servomotor to the specified pin and configure the trigger pin as an output and the echo pin as an input. We also initialize serial communication for debugging.
The loop() function repeatedly performs the following steps:

- Sends a brief pulse to the Ultrasonic sensor to trigger a distance measurement.
- Measures the duration of the echo pulse and calculates the distance in centimeters.
- Checks if an object is within the specified range (20 cm in this example) and, if so, opens the dustbin's flap by rotating the Servomotor to a specified angle (90 degrees).
- Prints the distance to the Serial Monitor for debugging purposes.
- Adds a delay between readings to prevent rapid triggering.

You can adjust the distance threshold, Servomotor angle, and delay times to match your specific setup and requirements.

Testing and Troubleshooting (30 minutes):

- Have students test their Smart Dustbin prototypes and troubleshoot any issues with the code or hardware.
- Encourage experimentation with different detection distances and servo motor angles.

Day 3

Project Presentation and Discussion (60 minutes):

- Each group presents their Smart Dustbin project to the class, explaining the design, components, and code.
- Discuss the environmental benefits of smart waste management systems and their potential impact on waste reduction.

Open Discussion and Future Improvements (30 minutes):

- Facilitate a class discussion on potential improvements to the Smart Dustbin and other applications of similar technology.
- Encourage students to brainstorm ideas for further enhancing waste management solutions.

Assessments

Assess students based on their participation, project functionality, understanding of Arduino programming, and their ability to troubleshoot issues.

Homework

Assign homework that requires students to research and present real-world examples of smart waste management systems and their impact on sustainability.

Conclusion

Conclude the lesson by summarizing the key concepts learned, highlighting the intersection of technology and environmental solutions, and encouraging students to think critically about applying technology for positive change.

English

FROM GAME TO STEM

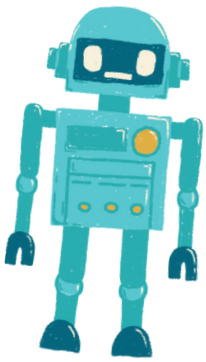
Co-funded by the
Erasmus+ Programme
of the European Union

Lesson 10
Arduino Ecology Lesson

Exploring Air Quality with Arduino

Duration:

1 periods



English

Objective:

- Students will learn about the importance of air quality in environmental health.
- Students will understand how air pollutants can impact ecosystems and human health.
- Students will build a simple Arduino-based air quality sensor and collect data.
- Students will discuss the significance of monitoring air quality for ecological well-being.

Materials:

- Arduino Uno boards (one per student or group)
- Air quality sensor module (e.g., MQ-135)
- Jumper wires
- USB cables for programming
- Computers with Arduino IDE installed
- Projector or whiteboard for demonstrations

Activities

Day 1

Introduction to Air Quality (10 minutes):

- Begin the lesson by discussing the importance of clean air and its impact on ecosystems.
- Explain how air pollutants can affect both natural environments and human health.

- Air Quality Sensors (20 minutes):

- Introduce air quality sensors and their role in monitoring air pollution.
- Explain the basic operation of air quality sensors and how they measure various gases.

Arduino Basics (10 minutes):

- Introduce Arduino as a microcontroller platform for data collection.
- Show students the components of an Arduino board and the basics of writing and uploading code.

Hands-on Activity Preparation (15 minutes):

- Provide each group with an Arduino Uno, an air quality sensor module, and jumper wires.
- Instruct students to assemble the air quality sensor and connect it to the Arduino.

Building and Programming the Air Quality Sensor (20 minutes):

- Guide students in building their Arduino-based air quality sensor system.
- Show students how to write Arduino code to collect air quality data from the sensor.

Here's a simple Arduino code example for monitoring air quality using an MQ-135 air quality sensor. This code reads data from the sensor and displays it on the Serial Monitor. Make sure you have the appropriate libraries installed in your Arduino IDE, as the MQ-135 sensor may require calibration for accurate results.

```
// Include necessary libraries
#include <Wire.h>
#include <Adafruit_Sensor.h>
#include <Adafruit_MQ135.h>

// Define the analog pin where the MQ-135 sensor is connected
#define MQ135_PIN A0

// Create an instance of the Adafruit MQ135 class
Adafruit_MQ135 mq135(MQ135_PIN);

void setup() {
 // Initialize serial communication for data output
 Serial.begin(9600);
}

void loop() {
 // Read the MQ-135 sensor's data
 float airQuality = mq135.readCO2();

 // Print the air quality data to the Serial Monitor
 Serial.print("Air Quality: ");
 Serial.print(airQuality);
 Serial.println(" ppm");

 // Add a delay before the next reading
 delay(2000); // Adjust the delay time as needed
}
```

In this code:

We include the necessary libraries, including the Adafruit Sensor and Adafruit MQ135 libraries, which are commonly used for working with the MQ-135 air quality sensor. Make sure you have these libraries installed in your Arduino IDE.

We define the analog pin (A0 in this example) where the MQ-135 sensor is connected to the Arduino.

An instance of the Adafruit_MQ135 class is created to interact with the sensor.

In the setup() function, we initialize serial communication for data output to the Serial Monitor.

In the loop() function, we continuously read the air quality data from the MQ-135 sensor using mq135.readCO2(). The data represents the CO2 concentration in parts per million (ppm).

The air quality data is printed to the Serial Monitor, and there is a delay of 2 seconds (adjustable) before the next reading is taken.

Please note that the MQ-135 sensor may require calibration for accurate readings, and the code provided here serves as a basic example. Depending on your specific application and calibration requirements, you may need to adjust the code and calibration process accordingly.

Data Collection and Discussion (20 minutes):

- Instruct students to collect air quality data by running their sensors in different environments (e.g., indoors, near a road, in a garden).
- Have students record and share their data with the class.
- Lead a class discussion on the differences in air quality data and the potential ecological consequences.

Assessments

Assess students based on their participation, data collection, and their ability to discuss the impact of air quality on ecological systems.

Homework

Assign homework that requires students to research and present real-world examples of ecological issues related to air pollution and the importance of air quality monitoring in mitigating these issues.

Conclusion

Conclude the lesson by summarizing the key ecological concepts learned, emphasizing the role of technology (Arduino) in environmental monitoring, and encouraging students to consider the significance of air quality for ecological well-being.

FROM GAME TO STEM

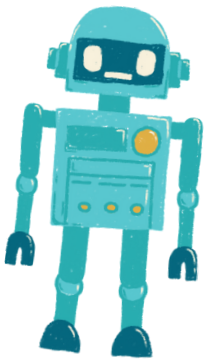
With the support of the Erasmus+ Programme of the European Union

Lección 1
Lección de Matemáticas con Arduino

Trigonometría con Arduino

Duración:

3 períodos



Objetivo:

- Los estudiantes comprenderán conceptos básicos de trigonometría, incluyendo el seno, coseno y tangente.
- Los estudiantes aplicarán funciones trigonométricas para resolver problemas del mundo real.
- Los estudiantes programarán un Arduino para crear un dispositivo simple de medición de ángu utilizando un motor servo.

Materiales:

- Placas de Arduino (una por estudiante o grupo)
- Motores servo
- Placas de pruebas (breadboards)
- Cables puente (jumper wires)
- Transportadores
- Reglas
- Cables USB para programación
- Computadoras con Arduino IDE instalado
- Proyector o pizarra para demostraciones

Actividades

Día 1

<u>Introducción a la Trigonometría (15 minutos):</u>

- Comience la lección presentando el concepto de trigonometría y su importancia en aplicaciones del mundo real.
- Explique brevemente el seno, el coseno y la tangente como razones de los lados en triángulos rectángulos.
- Hable sobre cómo las funciones trigonométricas pueden utilizarse para resolver problemas relacionados con ángulos y distancias.

<u>Fundamentos de la Trigonometría (30 minutos):</u>

- Profundice en las definiciones del seno, el coseno y la tangente.
- Proporcione ejemplos de cómo estas funciones se utilizan para encontrar ángulos faltantes o longitudes de lados en triángulos rectángulos.
- Permita que los estudiantes practiquen resolviendo problemas básicos de trigonometría en papel.

<u>Introducción a Arduino y Motores Servo (20 minutos):</u>

- Presente Arduino como una plataforma para crear dispositivos y explique sus componentes (microcontrolador, sensores, actuadores).
- Explique el concepto de motores servo y cómo se pueden controlar para moverse a ángulos específicos.
- Muestre ejemplos de control de motores servo con Arduino.

<u>Actividad Práctica (45 minutos):</u>

- Divida a los estudiantes en parejas o grupos pequeños.
- Proporcione a cada grupo una placa de Arduino, un motor servo, una placa de pruebas y cables puente.
- Instruya a los estudiantes para que ensamblen un dispositivo simple de medición de ángulos utilizando un motor servo y un transportador como referencia.
- Guíe a los estudiantes en la escritura de código Arduino para controlar el motor servo y hacerlo moverse a un ángulo específico según la entrada del usuario."

Día 2

Revisión de Conceptos de Trigonometría (20 minutos):

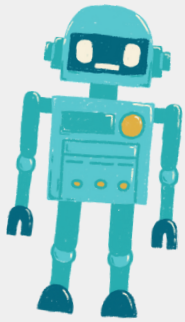
- Comience el segundo día revisando los conceptos de seno, coseno y tangent aprendidos en el Día 1.
- Proporcione problemas adicionales para que los estudiantes los resuelvan utilizando funciones trigonométricas.

Programación del Dispositivo de Medición de Ángulos (45 minutos):

- Continúe con la actividad práctica comenzada en el Día 1.
- Instruya a los estudiantes para que completen su código de Arduino para medir con precisión y mostrar ángulos utilizando el motor servo y una pantalla numérica (por ejemplo, Monitor Serial).
- Anime a los estudiantes a aplicar la trigonometría para convertir la posición del servo en ángulos.

Este es un ejemplo de código Arduino para crear un dispositivo simple de medición de ángulos utilizando un motor servo. Este código permite que el motor servo se mueva a un ángulo especificado según la entrada del usuario y muestra el ángulo medido en el Monitor Serie.

```
#include <Servo.h>
Servo myservo; // Create a Servo object
void setup() {
myservo.attach(9); // Attaches the servo to digital pin 9
Serial.begin(9600); // Initialize serial communication for debugging
}
void loop() {
int angle; // Variable to store the desired angle
Serial.println("Enter an angle (0-180): ");
while (!Serial.available()) {
  // Wait for user input
}
angle = Serial.parseInt(); // Read the angle entered by the user
if (angle >= 0 && angle <= 180) {
// Check if the entered angle is within the valid range
myservo.write(angle); // Move the servo to the specified angle
delay(500); // Delay for stability
Serial.print("Measured angle: ");
Serial.print(angle);
Serial.println(" degrees");
} else {
Serial.println("Invalid angle. Please enter a value between 0 and 180.");
}
}
```



Tu explicación proporciona un resumen claro y conciso del código de Arduino y cómo funciona para crear un dispositivo de medición de ángulos simple utilizando un motor servo. Describe los pasos clave y las funciones en el código, lo que facilita que los estudiantes lo entiendan y sigan.

De hecho, este proyecto es una excelente manera para que los estudiantes apliquen conceptos de trigonometría de manera práctica y atractiva, combinando las matemáticas con la electrónica y la programación práctica. Fomenta la resolución de problemas y el pensamiento crítico, al mismo tiempo que demuestra las aplicaciones del mundo real de la trigonometría en campos como la robótica y la ingeniería.

Día 3

Presentación del Proyecto y Discusión (60 minutos):

- Cada grupo presenta su dispositivo de medición de ángulos Arduino a la clase.
- Anime a los estudiantes a explicar cómo aplicaron conceptos de trigonometría en sus proyectos.
- Discuta las aplicaciones del mundo real de los dispositivos de medición de ángulos y la trigonometría.
- Facilite una discusión en clase sobre los desafíos y soluciones encontrados durante el proyecto."

Evaluciones

Evaluar a los estudiantes en función de su participación, calidad del código, precisión de la medición de ángulos y su capacidad para explicar los principios trigonométricos detrás de su dispositivo.

Tarea

Asigne una tarea para casa que requiera que los estudiantes investiguen y presenten una aplicación del mundo real de la trigonometría en diversos campos, como ingeniería, física o arquitectura.

Conclusión

Concluya la lección resumiendo los conceptos clave de trigonometría aprendidos y enfatizando su uso práctico en el desarrollo de dispositivos Arduino. Destaque la conexión entre las matemáticas y la tecnología.

FROM GAME TO STEM

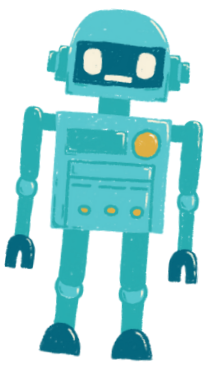
With the support of the Erasmus+ Programme of the European Union

Lección 2
Lección de Matemáticas con Arduino

Modelado Algebraico con Arduino

Duración:

2 períodos



Objetivo:

- Los estudiantes aprenderán a utilizar conceptos algebraicos para resolver problemas del mundo real.
- Los estudiantes aplicarán habilidades de modelado matemático y programación para crear un dispositivo Arduino que resuelve un problema práctico.
- Los estudiantes ganarán experiencia con variables y ecuaciones.

Materiales:

- Placas de Arduino (una por estudiante o grupo)
- Placas de pruebas (breadboards)
- Sensores o dispositivos de entr (por ejemplo, sensor de temperat sensor de luz, pulsador)
- LEDs, resistencias y cables puen (jumper wires)
- Cables USB para programación
- Computadoras con Arduino IDE instalado
- Proyector o pizarra para demostraciones

Actividades

Día 1

<u>Introducción al Modelado Algebraico (15 minutos):</u>

- Comience la lección presentando el modelado algebraico y su relevancia en la resolución de problemas del mundo real.
- Discuta la importancia de las variables y las ecuaciones en el modelado matemático.

<u>Variables y Ecuaciones (30 minutos):</u>

- Repase el concepto de variables y cómo se utilizan para representar valores desconocidos.
- Introduzca ecuaciones lineales (por ejemplo, y = mx + b) como una forma de modelar relaciones entre variables.
- Proporcione ejemplos de problemas del mundo real que se pueden representar mediante ecuaciones.

<u>Conceptos Básicos de Arduino (20 minutos):</u>

- Presente Arduino como una plataforma para crear dispositivos y explique sus componentes.
- Muestre ejemplos de proyectos simples de Arduino y cómo se pueden utilizar variables en la programación.

<u>Actividad Práctica (45 minutos):</u>

- Divida a los estudiantes en parejas o grupos pequeños.
- Proporcione a cada grupo una placa de Arduino, una placa de pruebas, un sensor o dispositivo de entrada (por ejemplo, un sensor de temperatura) y un LED.
- Instruya a los estudiantes para que creen un proyecto simple de Arduino que utilice un sensor para leer datos y un LED para representar visualmente los datos.
- Anime a los estudiantes a utilizar variables para almacenar lecturas del sensor y crear ecuaciones para controlar el LED en función de los datos del sensor.
- Discuta diferentes escenarios del mundo real en los que su dispositivo podría ser útil."

Día 2

Revisión de Conceptos Algebraicos (20 minutos):

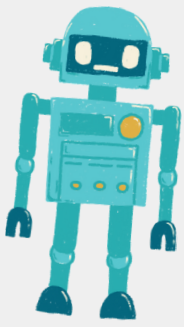
- Comience el segundo día revisando los conceptos de variables, ecuaciones modelado matemático.
- Discuta los proyectos de Arduino del día anterior y sus aplicaciones.

Programación del Modelo Algebraico (45 minutos):

- Continúe con la actividad práctica del Día 1.
- Instruya a los estudiantes para que modifiquen su código de Arduino y creen un modelo matemático utilizando datos del sensor.
- Anímelos a experimentar con diferentes ecuaciones y variables para representar la relación entre los datos del sensor y la salida del LED.
- Discuta cómo se puede utilizar el modelo para hacer predicciones o controlar sistemas del mundo real."

Aquí tienes un ejemplo de código de Arduino para un proyecto simple de modelado algebraico. En este proyecto, los estudiantes utilizarán un sensor de temperatura para leer datos de temperatura y controlar un LED en función de la lectura de temperatura. El código utiliza variables y una ecuación para determinar cuándo debe encenderse o apagarse el LED.

```
// Define the pins for the temperature sensor, LED, and a resistor (if needed)
const int temperatureSensorPin = A0; // Analog pin for the temperature sensor
const int ledPin = 13; // Use the built-in LED on most Arduino boards
const float thresholdTemperature = 25.0; // Define the threshold temperature
void setup() {
pinMode(ledPin, OUTPUT); // Set the LED pin as an output
Serial.begin(9600); // Initialize serial communication for debugging (optional)
}
void loop() {
// Read the temperature from the sensor
int sensorValue = analogRead(temperatureSensorPin);
// Convert the analog value to temperature in degrees Celsius
float temperatureCelsius = map(sensorValue, 0, 1023, 0, 100); // Adjust the mapping as
needed
// Check if the temperature is above the threshold
if (temperatureCelsius > thresholdTemperature) {
digitalWrite(ledPin, HIGH); // Turn the LED on
} else {
digitalWrite(ledPin, LOW); // Turn the LED off
}
// Print the temperature to the serial monitor for debugging (optional)
Serial.print("Temperature: ");
Serial.print(temperatureCelsius);
Serial.println(" °C");
// Delay for a moment to avoid rapid LED toggling
delay(1000); // Delay for 1 second
}
```



En este código:

La definición de los pines para el sensor de temperatura (conectado a un pin analógico), el LED (generalmente el LED incorporado en la mayoría de las placas Arduino) y el umbral de temperatura que determina cuándo debe encenderse el LED.
En la función `setup()`, se configura el pin del LED como una salida y se inicializa la comunicación serial para depuración (opcional).
En la función `loop()`, se lee la temperatura del sensor utilizando `analogRead()`. Luego, se convierte el valor analógico del sensor en temperatura en grados Celsius en función de las características del sensor.
Luego, se verifica si la temperatura supera el umbral definido (`thresholdTemperature`). Si la temperatura está por encima del umbral, se enciende el LED; de lo contrario, se apaga.
Opcionalmente, se imprime la temperatura en el Monitor Serie con fines de depuración.
Para utilizar este código, los estudiantes necesitarán conectar un sensor de temperatura (por ejemplo, LM35) a un pin analógico en el Arduino y un LED a un pin digital. El código leerá la temperatura del sensor y controlará el LED en función de la lectura de temperatura y el valor de umbral.

Presentación del Proyecto y Discusión (30 minutos):

- Cada grupo presenta su dispositivo Arduino a la clase.
- Anime a los estudiantes a explicar el modelo matemático que crearon y cómo funciona.
- Facilite una discusión en clase sobre las aplicaciones del modelado algebraico en la resolución de problemas prácticos."

Evaluaciones

Evaluar a los estudiantes en función de su participación, la funcionalidad de su dispositivo Arduino y su capacidad para explicar los conceptos algebraicos utilizados en su proyecto.

Tarea

Asigne una tarea para casa que requiera que los estudiantes investiguen y presenten un problema del mundo real que se pueda resolver utilizando el modelado algebraico y un dispositivo Arduino.

Conclusión

Concluya la lección resumiendo los conceptos clave de álgebra aprendidos y destacando la importancia del modelado matemático en la tecnología y la ingeniería.

FROM GAME TO STEM

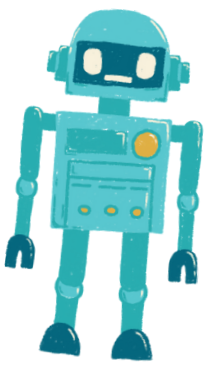
Co-funded by the Erasmus+ Programme of the European Union

Lección 3
Lección de Física con Arduino

Explorando el Movimiento y la Aceleración con Arduino

Duración:

3 períodos



Objetivo:

- Los estudiantes comprenderán los principios básicos del movimiento y la aceleración.
- Los estudiantes aplicarán ecuaciones cinemáticas para resolver problemas del mundo real relacionados con el movimiento.
- Los estudiantes programarán un Arduino para medir y mostrar la aceleración utilizando un sensor.

Materiales::

- Placas Arduino (una por estudiante o grupo)
- Módulos de sensor de acelerómetro (por ejemplo, ADXL345)
- Protoboards
- Cables de puente
- Cables USB para la programación
- Computadoras con el entorno de desarrollo Arduino IDE instalado
- Proyector o pizarra blanca para las demostraciones
- Opcional: Objetos de la vida real para experimentos de movimiento (por ejemplo, autos de juguete, pelotas)

Actividades

Día 1

Introducción al Movimiento y la Aceleración (15 minutos):

- Comienza la lección presentando el concepto de movimiento y aceleración.
- Define términos clave como velocidad, aceleración y desaceleración.
- Discute la importancia de comprender el movimiento en diversos campos, como la física, la ingeniería y el transporte.

Física del Movimiento (30 minutos):

- Explica las ecuaciones del movimiento, incluyendo:
- Desplazamiento: $s = ut + \frac{1}{2}at^2$
- Velocidad: $v = u + at$
- Aceleración: $a = \frac{v-u}{t}$
- Proporciona ejemplos de cómo se utilizan estas ecuaciones para analizar el movimiento.
- Realiza experimentos simples de movimiento (por ejemplo, rodar una pelota cuesta abajo) para demostrar los principios del movimiento.

Introducción a Arduino y Acelerómetros (20 minutos):

- Presenta Arduino como una plataforma para crear dispositivos y explica sus componentes.
- Explica el concepto de acelerómetros y cómo miden la aceleración.
- Muestra ejemplos de datos de acelerómetros y explica cómo se relacionan con el movimiento del mundo real.

Actividad Práctica (45 minutos):

- Divide a los estudiantes en parejas o grupos pequeños.
- Proporciona a cada grupo una placa Arduino, un módulo de sensor de acelerómetro, una protoboard y cables de puente.
- Instruye a los estudiantes para que ensamblen un circuito simple que conecte el acelerómetro a la placa Arduino.
- Guía a los estudiantes en la escritura de código Arduino para leer y mostrar datos de aceleración del sensor en el Monitor Serie.

Día 2

Revisión de las Ecuaciones Cinemáticas (20 minutos):

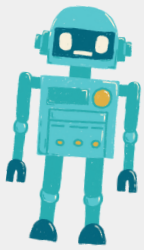
- Comienza el segundo día revisando las ecuaciones cinemáticas discutidas en el primer día.
- Proporciona ejemplos adicionales para que los estudiantes los resuelvan utilizando estas ecuaciones.

Programación del Dispositivo de Medición de Aceleración (45 minutos):

- Continúa con la actividad práctica iniciada en el primer día.
- Instruye a los estudiantes para que completen su código de Arduino y puedan leer y mostrar datos de aceleración en tiempo real del acelerómetro.
- Anima a los estudiantes a calibrar el sensor si es necesario y a aplicar las ecuaciones de aceleración para interpretar los datos.

Aquí tienes un ejemplo de código de Arduino para medir y mostrar la aceleración utilizando un sensor de acelerómetro ADXL345. Este código permitirá a tus estudiantes conectar el sensor de acelerómetro con Arduino y mostrar los valores de aceleración en el Monitor Serie.

```
#include <Wire.h>
#include <Adafruit_Sensor.h>
#include <Adafruit_ADXL345_U.h>
Adafruit_ADXL345_Unified accel = Adafruit_ADXL345_Unified(12345);
void setup(void) {
 Serial.begin(9600);
 Serial.println("Accelerometer Test"); Serial.println("");
  if(!accel.begin())
 {
/* There was a problem detecting the ADXL345 ... check your connections*/
   Serial.println("Ooops, no ADXL345 detected ... Check your wiring!");
   while(1);
 }
}
void loop(void) {
 sensors_event_t event;
 accel.getEvent(&event);
 /* Display the acceleration values (in m/s^2) */
 Serial.print("X: "); Serial.print(event.acceleration.x); Serial.print(" ");
 Serial.print("Y: "); Serial.print(event.acceleration.y); Serial.print(" ");
 Serial.print("Z: "); Serial.print(event.acceleration.z); Serial.println(" m/s^2");
 delay(500); // Delay for half a second between readings
}
```



En este código:
Incluimos las bibliotecas necesarias para el sensor de acelerómetro ADXL345.
Creamos un objeto Adafruit_ADXL345_Unified llamado "accel" para interactuar con el sensor.
En la función "setup()", inicializamos la comunicación serial para fines de depuración y verificamos si se detecta el sensor de acelerómetro. Si no se detecta, se mostrará un mensaje de error.
En la función "loop()", leemos continuamente los datos de aceleración del sensor utilizando "accel.getEvent(&event)" y mostramos los valores de aceleración en los ejes X, Y y Z en metros por segundo al cuadrado (m/s^2) en el Monitor Serie.
Para utilizar este código, asegúrate de que tus estudiantes hayan conectado correctamente el sensor de acelerómetro ADXL345 al Arduino. El sensor debe estar conectado a los pines apropiados (por ejemplo, SDA y SCL) para la comunicación I2C. Cuando el Arduino esté encendido y ejecutando este código, mostrará continuamente los datos de aceleración en el Monitor Serie.
Ten en cuenta que es posible que debas instalar la biblioteca Adafruit ADXL345 a través del Administrador de bibliotecas de Arduino para poder utilizar este código.

Presentación del Proyecto y Discusión (60 minutos):

- Cada grupo presenta su dispositivo de medición de aceleración basado en Arduino a la clase.
- Anima a los estudiantes a explicar cómo aplicaron las ecuaciones cinemáticas y los principios de la física en sus proyectos.
- Habla sobre las aplicaciones del mundo real de los dispositivos de medición de aceleración en diversos campos.
- Facilita una discusión en clase sobre los desafíos y soluciones encontrados durante el proyecto.

Evaluaciones

Evaluar a los estudiantes en función de su participación, calidad del código, precisión de la medición de aceleración y su capacidad para explicar los principios de la física detrás de su dispositivo.

Tarea

Asignar una tarea que requiera a los estudiantes investigar y presentar una aplicación del mundo real de la medición de aceleración en física o ingeniería.

Resumen

Concluyan la lección resumiendo los conceptos clave de física aprendidos y enfatizando su uso práctico en el desarrollo de dispositivos Arduino. Destaquen la conexión entre la física y la tecnología.

FROM GAME TO STEM

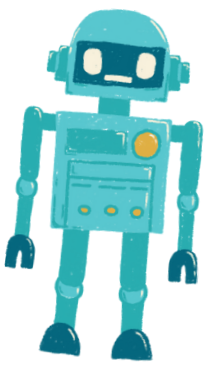
Co-funded by the Erasmus+ Programme of the European Union

Lección 4
Lección de Física con Arduino

Explorando Oscilaciones y Movimiento de Péndulo con Arduino

Duración:

2 períodos



Objetivo:

- Los estudiantes comprenderán los principios del movimiento oscilatorio y las oscilaciones armónicas.
- Los estudiantes aplicarán la modelización matemática para analizar el movimiento del péndulo.
- Los estudiantes programarán un Arduino para simular y visualizar el movimiento del péndulo.

Materiales::

- Placas Arduino (una por estudiante o grupo)
- Servomotores
- Protoboards
- Cables de puente
- Pequeños objetos como pesos de péndulo (por ejemplo, arandelas)
- Cuerda o hilo para péndulos
- Reglas o cinta métrica
- Cables USB para programación
- Computadoras con el entorno de desarrollo Arduino IDE instalado
- Proyector o pizarra blanca para demostraciones

Actividades

Día 1

Introducción a las Oscilaciones y el Movimiento del Péndulo (15 minutos):

- Comienza la lección presentando el concepto de movimiento oscilatorio y su relevancia en diversos campos, incluyendo la física y la ingeniería.
- Define términos clave como período, frecuencia, amplitud y oscilaciones armónicas.
- Explica los fundamentos del movimiento del péndulo y su representación matemática.

Modelado Matemático de Péndulos (30 minutos):

- Discute el modelo matemático de un péndulo simple, incluyendo la ecuación para el período de un péndulo: $T = 2\pi\sqrt{\frac{L}{g}}$
- donde T es el período, L es la longitud del péndulo y g es la aceleración debido a la gravedad.
- Proporciona ejemplos de cómo utilizar la ecuación para calcular el período de un péndulo.
- Realiza cálculos simples relacionados con el movimiento del péndulo.

Introducción a Arduino y los Servomotores (20 minutos):

- Presenta Arduino como una plataforma para crear dispositivos y explica sus componentes (microcontrolador, sensores, actuadores).
- Explica el concepto de servomotores y cómo se pueden utilizar para simular el movimiento del péndulo.
- Muestra ejemplos de control de servomotores con Arduino.

Actividad Práctica (45 minutos):

- Divide a los estudiantes en parejas o grupos pequeños.
- Proporciona a cada grupo una placa Arduino, un servomotor, una protoboard, cables de puente, un pequeño objeto como peso de péndulo y una cuerda.
- Instruye a los estudiantes para que ensamblen un simulador de péndulo simple utilizando el servomotor para controlar el movimiento del péndulo.
- Guía a los estudiantes en la escritura de código de Arduino para crear oscilaciones armónicas variando la posición del servomotor.

Día 2

Revisión del Movimiento del Péndulo (20 minutos):

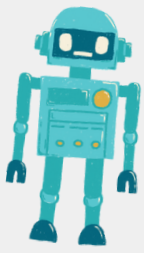
- Comienza el segundo día revisando los conceptos de movimiento oscilatorio, movimiento del péndulo y el modelo matemático de un péndulo simple.
- Discute los proyectos de Arduino del día anterior y sus aplicaciones.

Programación del Simulador de Péndulo (45 minutos):

- Continúa con la actividad práctica del primer día.
- Instruye a los estudiantes para que modifiquen su código de Arduino para simular con precisión el movimiento del péndulo con diferentes parámetros como longitud y amplitud.
- Anima a los estudiantes a visualizar y representar gráficamente el movimiento utilizando el servomotor.

Aquí tienes un ejemplo de código de Arduino para crear un simulador de péndulo simple utilizando un servomotor. Este código permite a los estudiantes programar un Arduino para simular y visualizar el movimiento de un péndulo:

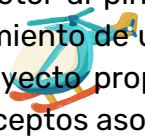
```
#include <Servo.h>
Servo pendulum; // Create a Servo object
int angle = 90; // Initial angle of the pendulum (straight down)
int amplitude = 45; // Maximum angle to one side of the vertical (adjust as needed)
int period = 2000; // Period of the pendulum swing in milliseconds (adjust as needed)
void setup() {
  pendulum.attach(9); // Attach the servo to digital pin 9
}
void loop() {
  // Calculate the angle of the pendulum using harmonic motion
  int displacement = amplitude * cos(2 * PI * millis() / period);
  int pendulumAngle = angle + displacement;
   // Move the servo to the calculated angle
  pendulum.write(pendulumAngle);
  // Delay for a short time to control the speed of the simulation
  delay(50); // Adjust as needed for desired speed
}
```



En este código:

1. Incluimos la biblioteca Servo para controlar el servomotor.
2. Creamos un objeto Servo llamado "pendulum" para controlar el servomotor.
3. Definimos variables para el ángulo inicial del péndulo (angle), el ángulo máximo hacia un lado de la vertical (amplitude) y el período del movimiento del péndulo (period). Puedes ajustar estos valores para cambiar el comportamiento del péndulo.
4. En la función "setup()", adjuntamos el servomotor al pin digital 9.
5. En la función "loop()", calculamos el ángulo del péndulo utilizando la fórmula para el movimiento armónico. El desplazamiento representa el desplazamiento angular desde la posición vertical, y lo sumamos al ángulo inicial para obtener "pendulumAngle".
6. Utilizamos "pendulum.write(pendulumAngle)" para mover el servomotor a un ángulo calculado.
7. Agregamos un retardo para controlar la velocidad de la simulación. Ajusta el valor del retardo según sea necesario para lograr la velocidad de simulación deseada.

Para utilizar este código, los estudiantes deben conectar un servomotor al pin digital 9 del Arduino y cargar el código en la placa. El servomotor simulará el movimiento de un péndulo, y los estudiantes podrán observar las oscilaciones en acción. Este proyecto proporciona una representación visual y práctica del movimiento del péndulo y sus conceptos asociados.



Presentación del Proyecto y Discusión (30 minutos):

- Cada grupo presenta su simulador de péndulo basado en Arduino a la clase.
- Anima a los estudiantes a explicar cómo aplicaron los principios de modelado matemático en sus proyectos.
- Discute las aplicaciones del mundo real de comprender las oscilaciones armónicas en campos como la física, la ingeniería y la astronomía.
- Facilita una discusión en clase sobre los desafíos y soluciones encontrados durante el proyecto.

Evaluaciones

Evalúa a los estudiantes en función de su participación, la calidad de su dispositivo Arduino y su capacidad para explicar los principios del movimiento oscilatorio y las oscilaciones armónicas.

Tarea

Tarea: Investigación y Presentación de Aplicaciones del Mundo Real de las Oscilaciones y el Movimiento Armónico.

Resumen

Concluyan la lección resumiendo los conceptos clave de física aprendidos y enfatizando su uso práctico en el desarrollo de dispositivos Arduino. Destaquen la conexión entre las matemáticas y la tecnología en la comprensión del movimiento del péndulo.

FROM GAME TO STEM

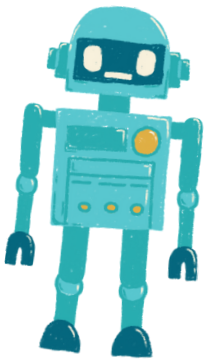
Co-funded by the Erasmus+ Programme of the European Union

Lección 5
Lección de Química con Arduino

Explorando la Hidroponía y la Química del Crecimiento de Plantas con Arduino

Duración:

3 períodos



Objetivo::

- Los estudiantes comprenderán los principios de la hidroponía y sus ventajas en el crecimiento de las plantas.
- Los estudiantes aprenderán sobre los nutrientes esenciales para el crecimiento de las plantas y cómo preparar soluciones nutritivas.
- Los estudiantes diseñarán y construirán un sistema de riego automatizado utilizando Arduino.
- Los estudiantes supervisarán y controlarán los niveles de pH en un sistema hidropónico para optimizar el crecimiento de las plantas.

Materiales:

- Placas Arduino (una por estudiante o grupo)
- Sensores de pH y soluciones de calibración de pH
- Bombas de agua
- Tubos y emisores de goteo
- Reservorio para la solución de nutrientes
- Soluciones de pH alto y pH bajo
- Soluciones de buffer de pH
- Semillas o plántulas de plantas
- Medio de cultivo hidropónico (por ejemplo, lana de roca, perlita)
- Componentes de la solución de nutrientes (por ejemplo, fertilizante N-P-K)
- Recipientes para configuraciones hidropónicas
- Cables de puente
- Cables USB para programación
- Computadoras con el entorno de desarrollo Arduino IDE instalado
- Proyector o pizarra blanca para demostraciones

Actividades

Día1

Introducción a la Hidroponía (15 minutos):

- Comienza la lección presentando el concepto de hidroponía y sus ventajas en el crecimiento de las plantas.
- Discute los beneficios de los ambientes controlados y los sistemas eficientes en el uso del agua.

Química del Crecimiento de las Plantas (30 minutos):

- Explica los elementos químicos esenciales para el crecimiento de las plantas (por ejemplo, nitrógeno, fósforo, potasio) y sus funciones.
- Habla sobre la importancia de los niveles de pH en la absorción de nutrientes y la salud de las plantas.
- Introduce el concepto de soluciones nutritivas y el control de pH en la hidroponía.

Introducción a Arduino y Sensores (20 minutos):

- Presenta Arduino como una plataforma para la automatización y la recopilación de datos, y explica sus componentes (microcontrolador, sensores).
- Muestra ejemplos de sensores utilizados en sistemas hidropónicos (por ejemplo, sensores de pH).

Preparación de la Actividad Práctica (45 minutos):

- Proporciona a los estudiantes placas Arduino, sensores de pH, bombas de agua, tubos y recipientes.
- Instruye a los estudiantes a que hagan una lluvia de ideas y planifiquen el diseño de su sistema hidropónico, incluyendo las soluciones nutritivas y el control de pH.

Día2

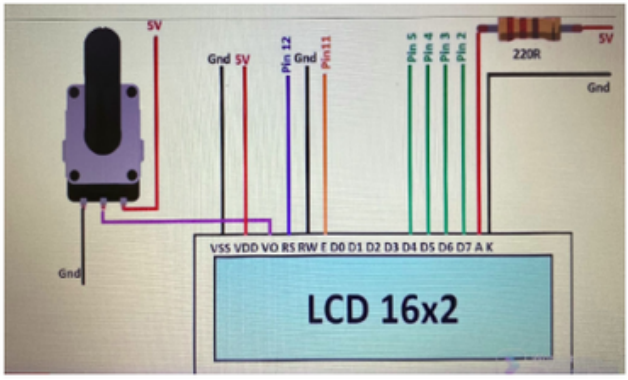
Configuración del Sistema Hidropónico (60 minutos):

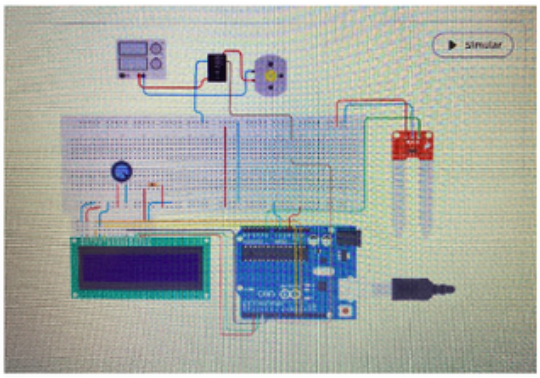
- Comienza el segundo día permitiendo que los estudiantes configuren sus sistemas hidropónicos.
- Los estudiantes deben plantar semillas o plántulas en un medio de cultivo hidropónico (por ejemplo, lana de roca) y organizar el sistema de riego con tubos y emisores de goteo.
- Demuestra cómo mezclar y preparar las soluciones nutritivas.

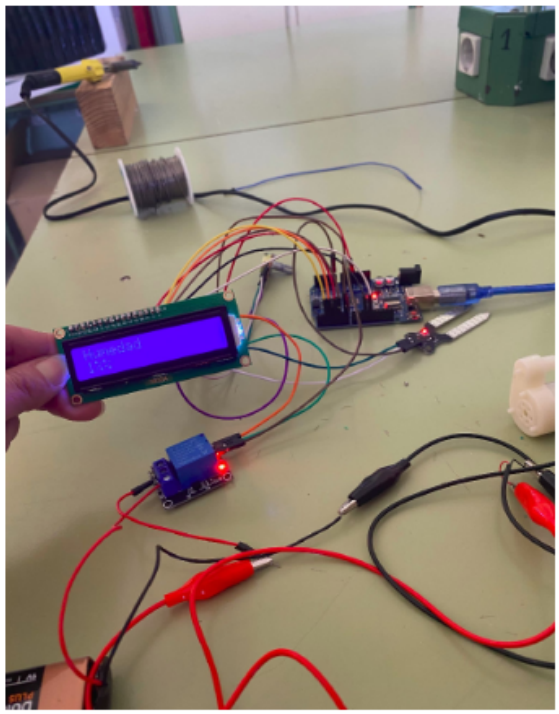
Monitoreo y Calibración de pH (30 minutos):

- Instruye a los estudiantes sobre cómo calibrar y utilizar los sensores de pH para el monitoreo de pH.
- Explica la importancia de mantener el pH dentro del rango óptimo para la absorción de nutrientes (generalmente alrededor de pH 6-7).
- Guía a los estudiantes en la calibración de sus sensores de pH utilizando soluciones buffer de pH.

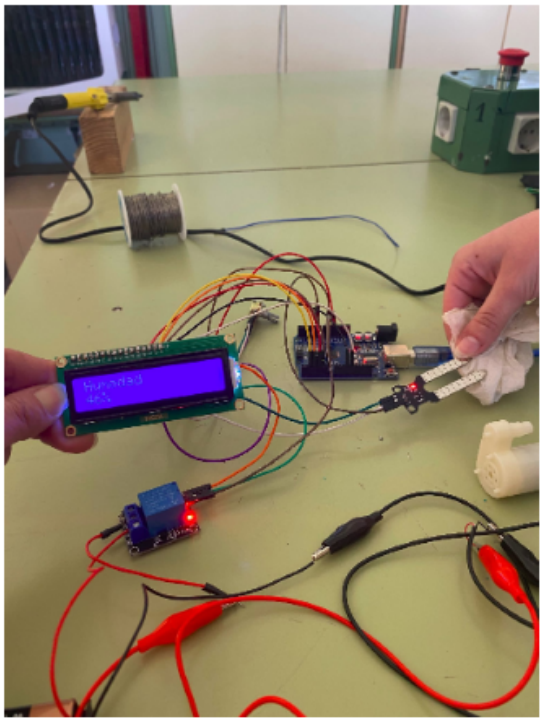
Diagrama de las Conexiones:







Cuando la humedad está al 1%, la bomba funciona.



Cuando la humedad está al 46%, la bomba sigue funcionando.

Día3

Programación de Arduino (60 minutos):

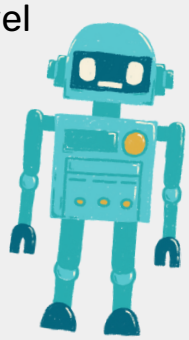
- Instruye a los estudiantes para que escriban código de Arduino para el sistema de rieg automatizado.
- Los estudiantes deben programar el Arduino para controlar el horario de la bomba de agua y monitorear los niveles de pH.
- Enfatiza la importancia del riego regular y los ajustes de pH para el crecimiento de las plantas.

Recopilación de Datos y Control de pH (30 minutos):

- Explica cómo recopilar datos de los sensores de pH y monitorear los niveles de pH.
- Discute las acciones que deben tomar los estudiantes si los niveles de pH se desvían del rango óptimo.

Aquí tienes un ejemplo de código de Arduino para un sistema de riego hidropónico automatizado con monitoreo y control de pH. Este código es un marco básico que puedes adaptar y ampliar según tu configuración y necesidades específicas.

```
#include <Adafruit_ADS1015.h>
#include <Wire.h>
#define PH_SENSOR_PIN 0 // Analog pin for pH sensor
#define PUMP_PIN 12    // Digital pin for water pump
// Create an Adafruit ADS1015 ADC object
Adafruit_ADS1015 ads;
float currentpH;
float targetpH = 6.5; // Adjust this value to your desired pH level
void setup() {
 Serial.begin(9600);
 ads.begin();
 pinMode(PUMP_PIN, OUTPUT);
 digitalWrite(PUMP_PIN, LOW); // Initialize pump as off
}
void loop() {
 // Read pH value from pH sensor
 currentpH = readpH();
 // Display current pH value
 Serial.print("Current pH: ");
 Serial.println(currentpH);
 // Check and adjust pH level
 if (currentpH < targetpH) {
  // pH is too low, add pH up solution (adjust as needed)
  // Implement pH adjustment mechanism here
    // For example, you can control a peristaltic pump for adding pH up solution
```



```
digitalWrite(PUMP_PIN, HIGH); // Turn on the pump
 delay(1000); // Adjust the delay time as needed
 digitalWrite(PUMP_PIN, LOW); // Turn off the pump
 } else if (currentpH > targetpH) {
 // pH is too high, add pH down solution (adjust as needed)
 // Implement pH adjustment mechanism here
  // For example, you can control a peristaltic pump for adding pH down solution
 digitalWrite(PUMP_PIN, HIGH); // Turn on the pump
 delay(1000); // Adjust the delay time as needed
 digitalWrite(PUMP_PIN, LOW); // Turn off the pump
 }
 // Add code for irrigation control here (e.g., based on time intervals)
}
float readpH() {
 // Read pH value from the pH sensor and convert to pH scale
 int16_t rawValue = ads.readADC_SingleEnded(PH_SENSOR_PIN);
 float voltage = (rawValue * 0.1875) / 1000.0; // Convert to voltage
  float pHValue = 3.5 * voltage + 3.5; // Convert to pH scale (adjust calibration values as needed)
 return pHValue;
}
```

En este código:

- Utilizamos la biblioteca Adafruit ADS1015 para comunicarnos con el convertidor analógico-digital ADS1015, que se utiliza para leer los valores del sensor de pH. Asegúrate de tener esta biblioteca instalada en tu entorno de desarrollo de Arduino.
- La función setup() inicializa la comunicación serial para la salida de datos, inicializa el ADC y configura el pin de la bomba de agua como salida.
- En la función loop(), leemos continuamente el valor de pH del sensor de pH utilizando la función readpH().
- Comparamos el valor de pH actual con el nivel de pH objetivo (targetpH) y ajustamos el pH según sea necesario. Debes implementar el mecanismo de ajuste de pH según tu configuración específica, que puede implicar el control de una bomba peristáltica para agregar soluciones de pH alto o pH bajo.
- Además, puedes agregar código para controlar la bomba de agua para el riego según intervalos de tiempo u otros criterios.

Ten en cuenta que este código proporciona un marco básico y es posible que debas calibrarlo y ajustarlo según tu configuración específica de sensor y bomba, así como los niveles de pH deseados y el programa de riego. Asegúrate de tomar precauciones de seguridad al trabajar con productos químicos y bombas.

Día4

Configuración del Experimento y Monitoreo (60 minutos):

- Permite que los estudiantes configuren sus sistemas hidropónicos automatizados en el invernadero o el aula.
- Los estudiantes deben poner en marcha el sistema y supervisar el crecimiento de las plantas, los niveles de nutrientes y el pH.

Presentación del Proyecto y Discusión (60 minutos):

- Cada grupo presenta su diseño de sistema hidropónico, metodología y resultados iniciales a la clase.
- Discute el impacto de las soluciones nutritivas y el control de pH en el crecimiento de las plantas.
- Anima a los estudiantes a proponer optimizaciones para sus sistemas basadas en observaciones iniciales.

Evaluaciones

Evalúa a los estudiantes en función de su participación, configuración del sistema, calibración de pH, recopilación de datos, análisis y la calidad de sus presentaciones.

Tarea

Asigna la siguiente Tarea que requiere que los estudiantes investiguen y presenten una aplicación del mundo real de la hidroponía en la agricultura o la agricultura sostenible.

Conclusión

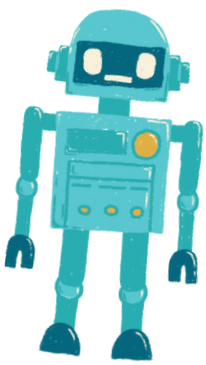
Concluimos esta lección de química reforzando los conceptos clave aprendidos y destacando la importancia de la hidroponía en la agricultura sostenible y la producción de alimentos. También resaltamos el papel fundamental de la tecnología, en particular Arduino, en la automatización y optimización de los sistemas hidropónicos.

Lección 6
Lección de Química con Arduino

Explorando Reacciones Químicas con Arduino

Duración:

3 períodos



Objetivo:

- Los estudiantes comprenderán los principios de las reacciones químicas y su relevancia en la vida cotidiana.
- Los estudiantes diseñarán y llevarán a cabo experimentos para observar y medir reacciones químicas.
- Los estudiantes programarán un sistema de monitoreo de temperatura basado en Arduino para recopilar y analizar datos de reacciones químicas.

Materiales:

- Placas Arduino (una por estudiante o grupo)
- Sensores de temperatura (por ejemplo, DS18B20 o LM35)
- Productos químicos para experimentos (por ejemplo, bicarbonato de sodio, vinagre)
- Recipientes para reacciones (por ejemplo, vasos de precipitados, tubos de ensayo)
- Cables de puente
- Cables USB para programación
- Computadoras con el entorno de desarrollo Arduino IDE instalado
- Proyector o pizarra blanca para demostraciones
- Gafas de seguridad y batas de laboratorio

Actividades

Día1

Introducción a las Reacciones Químicas (15 minutos):

- Comienza la lección presentando el concepto de reacciones químicas y su papel en la vida diaria.
- Discute la importancia de las reacciones químicas en diversos campos, incluyendo la química, la biología y la industria.

Conceptos Básicos de las Reacciones Químicas (30 minutos):

- Profundiza en los fundamentos de las reacciones químicas, incluyendo reactantes, productos y la conservación de la masa.
- Proporciona ejemplos de reacciones químicas comunes y sus aplicaciones.
- Enfatiza los protocolos de seguridad al manipular productos químicos en experimentos.

Introducción a Arduino y Sensores (20 minutos):

- Presenta Arduino como una plataforma para la recopilación de datos y explica sus componentes (microcontrolador, sensores).
- Muestra ejemplos de sensores de temperatura y cómo se pueden conectar a Arduino para la recopilación de datos.

Preparación de la Actividad Práctica (45 minutos):

- Proporciona a los estudiantes placas Arduino, sensores de temperatura, productos químicos y recipientes para reacciones.
- Instruye a los estudiantes a que hagan una lluvia de ideas y planifiquen sus experimentos de reacciones químicas, centrándose en los cambios de temperatura.
- Discute las precauciones de seguridad y las normas de laboratorio.

Día2

Configuración del experimento y recopilación de datos (60 minutos):

- Comience el segundo Día permitiendo a los estudiantes configurar sus experimento de reacciones químicas.
- Los estudiantes deben mezclar los químicos seleccionados en recipientes de reacción y colocar los sensores de temperatura.
- Indique a los estudiantes que escriban código Arduino para monitorear la temperatura y recopilar datos.

Análisis y discusión de datos (30 minutos):

- Guíe a los estudiantes en el análisis de sus datos, buscando cambios de temperatura durante las reacciones químicas.
- Discuta la importancia de los cambios de temperatura en las reacciones químicas y los conceptos de reacciones exotérmicas y endotérmicas.
- Anime a los estudiantes a sacar conclusiones basadas en sus hallazgos.

Día3

Programación del Sistema de Monitoreo (60 minutos):

- Instruya a los estudiantes a ajustar su código Arduino para la recopilación y el análisis de datos.
- Los estudiantes deben programar el Arduino para registrar datos de temperatura a intervalos específicos y mostrarlos gráficamente (por ejemplo, en una pantalla LCD o en el monitor serie).

Presentación y discusión del proyecto (60 minutos):

- Cada grupo presenta a la clase la configuración, la metodología y los resultados de su experimento de reacción química.
- Anime a los estudiantes a explicar sus observaciones y las implicaciones de los cambios de temperatura en las reacciones químicas.
- Facilite una discusión en clase sobre las aplicaciones del mundo real de reacciones químicas en diversos campos.

A continuación se muestra un ejemplo de código Arduino que puede utilizar para controlar la temperatura durante un experimento de reacción química. Este código lee datos de temperatura de un sensor de temperatura DS18B20 y los muestra en el monitor serie. Puede personalizar y ampliar este código para adaptarlo a su experimento específico.

```
#include <OneWire.h>
#include <DallasTemperature.h>
// Data wire is plugged into digital pin 2
#define ONE_WIRE_BUS 2
// Create a OneWire object
OneWire oneWire(ONE_WIRE_BUS);
// Pass the OneWire reference to Dallas Temperature sensor
DallasTemperature sensors(&oneWire);
```

```
void setup() {
 // Initialize serial communication for data output
 Serial.begin(9600);

 // Start the temperature sensor library
 sensors.begin();
}

void loop() {
 // Request temperature readings
 sensors.requestTemperatures();

 // Read temperature in Celsius
 float temperatureC = sensors.getTempCByIndex(0);

 // Display temperature on the Serial Monitor
 Serial.print("Temperature: ");
 Serial.print(temperatureC);
 Serial.println(" °C");

 // Add code here for data storage or further analysis

 // Delay before the next temperature reading (adjust as needed)
 delay(1000); // 1-second delay
}
```

En este código:

Usamos la biblioteca Dallas Temperature para interactuar con el sensor de temperatura DS18B20. Asegúrese de tener esta biblioteca instalada en su IDE de Arduino.

La función setup() inicializa la comunicación serial para la salida de datos e inicia la biblioteca de sensores de temperatura.
En la función loop(), el programa solicita y lee continuamente datos de temperatura del sensor usando sensors.getTempCByIndex(0). El 0 indica el primer (y en este caso, el único) sensor de temperatura conectado.
Los datos de temperatura se muestran en el monitor serie con un retraso de 1 segundo entre lecturas. Puede ajustar el tiempo de retraso para controlar la frecuencia de recopilación de datos.

Puede modificar este código para incluir sensores adicionales para diferentes experimentos, implementar el almacenamiento de datos en una tarjeta SD o crear representaciones gráficas de los datos en una pantalla LCD, según sus requisitos específicos.

Evaluaciones

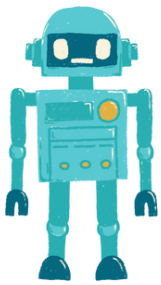
Evaluar a los estudiantes en función de su participación en el experimento, la recopilación de datos, el análisis y la calidad de sus presentaciones.

Tarea

Asigne Tarea que requiera que los estudiantes investiguen y presenten una aplicación del mundo real de reacciones químicas en una industria o campo de la ciencia específico.

Conclusión

Concluya la lección resumiendo los conceptos clave de química aprendidos y reforzando la importancia de las reacciones químicas para comprender los procesos naturales y los avances tecnológicos. Destacar el papel de la tecnología (Arduino) en las investigaciones científicas.



FROM GAME TO STEM

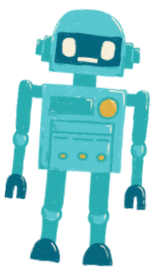
Co-funded by the Erasmus+ Programme of the European Union

Lección 7
Lección de Biología con Arduino

Investigación de la Conductividad Térmica con Termómetros Arduino

Duración:

3 períodos



Objetivo:

- Los estudiantes comprenderán el concepto de conductividad térmica y su importancia.
- Los estudiantes diseñarán y llevarán a cabo un experimento para determinar la conductividad térmica de diferentes materiales.
- Los estudiantes utilizarán termómetros Arduino para recopilar datos de temperatura y analizar sus hallazgos.

Materiales:

- Placas Arduino (una por estudiante o grupo)
- Sensores de temperatura (por ejemplo, DS18B20 o LM35)
- Placas de pruebas (breadboards)
- Cables de puente (jumper wires)
- Diversos materiales para probar la conductividad (por ejemplo, metales, plásticos, madera, vidrio)
- Materiales aislantes (por ejemplo, espuma, tela)
- Agua caliente o una fuente de calor
- Agua fría o hielo
- Cronómetro o temporizador
- Cables USB para programación
- Computadoras con el entorno de desarrollo Arduino IDE instalado
- Proyector o pizarra blanca para demostraciones
- Gafas de seguridad y guantes (para manipular materiales calientes)

Actividades

Día 1

Introducción a la Conductividad Térmica (15 minutos):

- Comienza la lección presentando el concepto de conductividad térmica y por qué es importante en la vida cotidiana.
- Discute aplicaciones del mundo real de la conductividad térmica, como en la cocina, los materiales de construcción y la ingeniería.

Transferencia de Calor y Aislamiento (30 minutos):

- Explica los diferentes métodos de transferencia de calor (conducción, convección, radiación) y concéntrate en la conducción, que es relevante para el experimento.
- Habla sobre los materiales aislantes y su papel en la reducción de la transferencia de calor.
- Enfatiza la necesidad de un experimento controlado para medir la conductividad térmica.

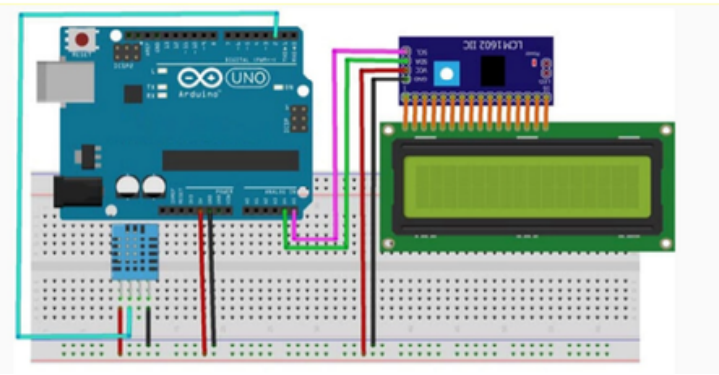
Introducción a los Termómetros Arduino (20 minutos):

- Presenta Arduino como una plataforma para la recopilación de datos y explica sus componentes (microcontrolador, sensores).
- Explica el uso de los sensores de temperatura y cómo se pueden conectar a las placas Arduino.
- Muestra ejemplos de recopilación y visualización de datos de temperatura con Arduino.

Preparación de la Actividad Práctica (45 minutos):

- Proporciona a los estudiantes placas Arduino, sensores de temperatura, placas de pruebas y cables de puente.
- Explica la configuración experimental: cada grupo probará diferentes materiales para determinar su conductividad térmica.
- Instruye a los estudiantes a que hagan una lluvia de ideas y planifiquen sus experimentos, incluyendo cómo controlar variables y recopilar datos.

ESQUEMA:



Día 2

Configuración del Experimento y Recopilación de Datos (60 minutos):

- Comienza el segundo día permitiendo que los estudiantes configuren sus experimentos.
- Los estudiantes deben conectar el sensor de temperatura a los materiales seleccionados y preparar contenedores de agua caliente y fría.
- Instruye a los estudiantes a que comiencen a recopilar datos de temperatura utilizando Arduino y registren las temperaturas iniciales.
- Pide a los estudiantes que midan y registren el tiempo que tarda en cambiar la temperatura en cada material.

Análisis de Datos y Discusión (30 minutos):

- Guía a los estudiantes en el análisis de sus datos, el cálculo de la tasa de cambio de temperatura y la comparación de la conductividad de diferentes materiales.
- Discute el concepto de resistencia térmica y cómo se relaciona con la conductividad.
- Anima a los estudiantes a identificar cuál es el mejor conductor de calor y cuál es el peor basándose en sus hallazgos.

Aquí tienes un ejemplo de código Arduino que puedes usar para el experimento de medición y comparación de la conductividad térmica de diferentes materiales utilizando un sensor de temperatura (por ejemplo, DS18B20). Este código recopilará datos de temperatura del sensor y te permitirá calcular la tasa de cambio de temperatura para cada material.

```
#include <OneWire.h>
#include <DallasTemperature.h>
// Data wire is plugged into digital pin 2
#define ONE_WIRE_BUS 2
// Create a OneWire object
OneWire oneWire(ONE_WIRE_BUS);
// Pass the OneWire reference to Dallas Temperature sensor
DallasTemperature sensors(&oneWire);
void setup() {
 // Initialize serial communication for data output
 Serial.begin(9600);
 // Start the temperature sensor library
 sensors.begin();
}
void loop() {
```

```
// Initialize variables for temperature measurements
 float initialTemp, finalTemp;
 unsigned long startTime, endTime;
 float rateOfChange;
 // Wait for the user to start the experiment (e.g., press a button)
while (!digitalRead(3)) {
  // Wait for button press
 }
 delay(1000); // Debounce delay
 // Measure initial temperature
 sensors.requestTemperatures(); // Request temperature readings
 initialTemp = sensors.getTempCByIndex(0); // Get temperature in Celsius
 // Record the start time
 startTime = millis();
 // Wait for the temperature to stabilize (e.g., 5 seconds)
 delay(5000);
 // Measure final temperature
 sensors.requestTemperatures();
 finalTemp = sensors.getTempCByIndex(0);
 // Record the end time
 endTime = millis();
 // Calculate the rate of temperature change (°C per second)
  rateOfChange = (finalTemp - initialTemp) / ((endTime - startTime) /
1000.0);
 // Output results to the Serial Monitor
 Serial.print("Initial Temperature: ");
 Serial.print(initialTemp);
 Serial.println(" °C");
 Serial.print("Final Temperature: ");
 Serial.print(finalTemp);
 Serial.println(" °C");
 Serial.print("Rate of Change: ");
 Serial.print(rateOfChange);
 Serial.println(" °C/s");
 // Wait for user input (e.g., press a button) to proceed to the next material
 Serial.println("Press a button to test the next material.");
 while (!digitalRead(3)) {
  // Wait for button press
 }
 delay(1000); // Debounce delay
}
```

En este código:
Utilizamos la biblioteca Dallas Temperature para interactuar con el sensor de temperatura DS18B20. Asegúrate de tener esta biblioteca instalada en tu entorno de desarrollo de Arduino.
Definimos el pin digital (por ejemplo, pin 2) al que está conectado el cable de datos del sensor DS18B20.
En la función setup(), inicializamos la comunicación serial para la salida de datos y comenzamos la biblioteca del sensor de temperatura.
En la función loop(), el programa espera a que se presione un botón para iniciar el experimento. Se mide la temperatura antes y después de un período de espera (por ejemplo, 5 segundos) para calcular la tasa de cambio de temperatura.
La tasa de cambio se calcula como la diferencia de temperatura dividida por el tiempo necesario para alcanzar ese cambio. Esto te proporciona la tasa en °C por segundo.
Los resultados se imprimen en el Monitor Serie, incluyendo la temperatura inicial, la temperatura final y la tasa de cambio.
Después de que se complete el experimento para un material, puedes presionar un botón para proceder al siguiente material.
Recuerda conectar correctamente el sensor DS18B20 al Arduino y asegurarte de que el botón esté conectado a un pin digital (por ejemplo, pin 3) para activar el experimento. Ajusta el código según sea necesario en función de tu configuración específica.

Día 3

Presentación del Proyecto y Discusión (60 minutos):

- Cada grupo presenta su configuración experimental, metodología y resultados a la clase.
- Anima a los estudiantes a explicar sus observaciones, discutir posibles fuentes de error y sugerir mejoras.
- Facilita una discusión en clase sobre las implicaciones del mundo real de sus hallazgos, como la selección de materiales para el aislamiento térmico o materiales conductores para disipadores de calor.

Evaluaciones

Evalúa a los estudiantes en función de su participación en el experimento, la recopilación de datos, el análisis y la calidad de sus presentaciones.

Tarea

Asigna una tarea que requiera a los estudiantes investigar y presentar una aplicación del mundo real de la conductividad térmica en la ingeniería o la ciencia de materiales.

Conclusión

Concluye la lección resumiendo los conceptos clave aprendidos y reforzando la importancia de comprender la conductividad térmica en la vida cotidiana y las aplicaciones tecnológicas.

FROM GAME TO STEM

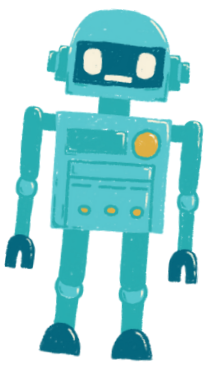
Co-funded by the Erasmus+ Programme of the European Union

Lección 8
Lección de Biología con Arduino

Explorando la Fotosíntesis y el Crecimiento de las Plantas con Arduino

Duración:

3 períodos



Objetivo:

- Los estudiantes comprenderán el proceso de la fotosíntesis y su importancia en el crecimiento de las plantas.
- Los estudiantes diseñarán y llevarán a cabo un experimento para investigar el efecto de los factores ambientales en la fotosíntesis.
- Los estudiantes programarán un sistema basado en Arduino para monitorear y recopilar datos relacionados con el crecimiento de las plantas.

Materiales:

- Placas Arduino (una por estudiante o grupo)
- Sensores (por ejemplo, sensor de luz, sensor de temperatura, sensor de humedad)
- Luces LED de crecimiento (opcional)
- Plantas en macetas o semillas
- Tierra y contenedores para plantar
- Cables de puente
- Cables USB para programación
- Computadoras con Arduino IDE instalado
- Proyector o pizarra blanca para demostraciones
- Tierra para macetas, agua y otros materiales para el cuidado de las plantas

Actividades

Día 1

<u>Introducción a la Fotosíntesis (15 minutos):</u>

- Comienza la lección presentando el concepto de fotosíntesis y su importancia en el crecimiento de las plantas.
- Discute la ecuación química de la fotosíntesis y el papel de la luz, el dióxido de carbono y el agua en el proceso.

<u>Factores Ambientales y Crecimiento de las Plantas (30 minutos):</u>

- Explica cómo varios factores ambientales, como la intensidad lumínica, la temperatura y la humedad, pueden afectar la fotosíntesis y el crecimiento de las plantas.
- Habla sobre por qué estos factores son esenciales para el desarrollo saludable de las plantas.
- Enfatiza la necesidad de realizar experimentos controlados para estudiar estos factores.

<u>Introducción a Arduino y Sensores (20 minutos):</u>

- Presenta Arduino como una plataforma para la recopilación de datos y explica sus componentes (microcontrolador, sensores).
- Muestra ejemplos de sensores comúnmente utilizados en la monitorización ambiental y el cuidado de las plantas.
- Explica el papel de los sensores en la recopilación de datos para experimentos.

<u>Preparación de la Actividad Práctica (45 minutos):</u>

- Proporciona a los estudiantes placas Arduino, sensores (por ejemplo, sensor de luz, sensor de temperatura, sensor de humedad) y luces LED de crecimiento (opcional).
- Instruye a los estudiantes a que realicen lluvias de ideas y planifiquen sus experimentos de crecimiento de plantas, centrándose en un factor ambiental.
- Pide a los estudiantes que preparen macetas con tierra y planten semillas o pequeñas plantas en macetas.

Día 2

Configuración del Experimento y Recopilación de Datos (60 minutos):

- Comienza el segundo día permitiendo que los estudiantes configuren sus experimentos.
- Los estudiantes deben colocar los sensores en el entorno de las plantas, conectarlos a Arduino y posicionar las luces LED de crecimiento (si las están utilizando).
- Instruye a los estudiantes a recopilar datos relacionados con el factor ambiental elegido, como la intensidad lumínica o la temperatura.
- Enfatiza la importancia de registrar datos de manera precisa y regular.

Análisis de Datos y Discusión (30 minutos):

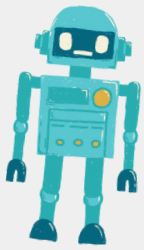
- Guía a los estudiantes en el análisis de sus datos, buscando tendencias o patrones relacionados con el crecimiento de las plantas y el factor elegido.
- Discute el impacto del factor en la fotosíntesis y el desarrollo de las plantas.
- Anima a los estudiantes a sacar conclusiones a partir de sus hallazgos.

Día 3

Programming the Monitoring System (60 minutes):

- Instruct students to write Arduino code to monitor and collect data from the sensors.
- Students should program the Arduino to record data at specific intervals (e.g., every hour).
- Discuss how to use libraries for sensor data retrieval and how to store data.

Aquí tienes un ejemplo de código Arduino para un sistema simple de monitorización ambiental que mide y registra datos relacionados con la intensidad de la luz utilizando un sensor de luz. Puedes adaptar este código para otros factores ambientales según sea necesario:



```
#include <Wire.h>
#include <Adafruit_Sensor.h>
#include <Adafruit_TSL2591.h>
Adafruit_TSL2591 tsl = Adafruit_TSL2591(2591);
void setup() {
 Serial.begin(9600);
  // Initialize the light sensor
 if(!tsl.begin()) {
   Serial.println("Light sensor not found. Check wiring.");
   while(1);
 }
 tsl.setGain(TSL2591_GAIN_LOW);    // Adjust the gain (options: LOW, MED, HIGH, MAX)
 tsl.setTiming(TSL2591_INTEGRATIONTIME_100MS); // Adjust the integration time (options: 100, 200, 300, 400, 500, 600)
}
void loop() {
 // Read and print light intensity data
 uint16_t luminance = tsl.getLuminosity(TSL2591_VISIBLE);
 Serial.print("Light Intensity (Lux): ");
 Serial.println(luminance);
    // Add code here to log data to an SD card, display on an LCD, or transmit to a computer/server.
 // Wait for a specific time interval (e.g., 1 hour)
 delay(3600000); // Adjust the delay time as needed
}
```

En este código:

Utilizamos la biblioteca Adafruit TSL2591 para interactuar con el sensor de luz TSL2591. Asegúrate de tener esta biblioteca instalada en tu entorno de desarrollo de Arduino.
La función setup() inicializa la comunicación serial para la salida de datos y configura el sensor de luz. También configura la ganancia y el tiempo de integración del sensor según tus necesidades.
En la función loop(), el programa lee continuamente los datos de intensidad de luz (en lux) del sensor utilizando tsl.getLuminosity(TSL2591_VISIBLE).
Puedes agregar código dentro del bucle para registrar los datos en una tarjeta SD, mostrarlos en una pantalla LCD o transmitirlos a una computadora o servidor para un análisis posterior. Por ejemplo, puedes utilizar un módulo de tarjeta SD para almacenar los datos localmente.
El retardo al final del bucle está configurado para esperar un intervalo de tiempo específico (por ejemplo, 1 hora) antes de tomar la siguiente lectura. Puedes ajustar el tiempo de retardo para controlar la frecuencia de recopilación de datos.

Este código proporciona un marco básico para monitorear la intensidad de luz, y puedes ampliarlo agregando más sensores para monitorear factores ambientales adicionales como la temperatura y la humedad. Recuerda ajustar el código para que coincida con los sensores y el hardware específicos que estás utilizando en tu experimento.

Presentación del Proyecto y Discusión (60 minutos):

- Cada grupo presenta su experimento de crecimiento de plantas, la configuración, la metodología y los resultados a la clase.
- Anima a los estudiantes a explicar sus observaciones y las implicaciones de sus hallazgos para el cuidado de las plantas y la agricultura.
- Facilita una discusión en clase sobre la importancia de la fotosíntesis en el ecosistema y cómo la tecnología (Arduino) puede ayudar en las investigaciones científicas.

Evaluaciones

Evalúa a los estudiantes en función de su participación en el experimento, la recopilación de datos, el análisis y la calidad de sus presentaciones.

Tarea

Asigna una tarea que requiera a los estudiantes investigar y presentar una aplicación del mundo real de la fotosíntesis y la monitorización ambiental en la agricultura o la ecología.

Conclusión

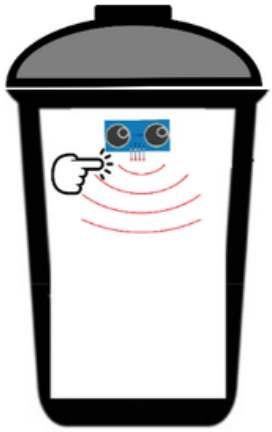
En esta lección, hemos explorado el fascinante mundo de la fotosíntesis y su importancia en el crecimiento de las plantas. Los estudiantes han tenido la oportunidad de diseñar y llevar a cabo experimentos para investigar cómo factores ambientales como la luz, la temperatura y la humedad afectan la fotosíntesis y, en última instancia, la salud de las plantas.

FROM GAME TO STEM

Co-funded by the Erasmus+ Programme of the European Union

Lección 9
Lección de Ecología de Arduino

Construyendo un Smart Dustbin con Arduino



Duración:

3 períodos

Objetivo::

- Los estudiantes comprenderán los conceptos básicos de la programación de Arduino y sus aplicaciones en la automatización.
- Los estudiantes aprenderán sobre los sensores ultrasónicos y cómo se pueden utilizar para la detección de objetos.
- Los estudiantes construirán un prototipo de Smart Dustbin y lo programarán para que abra su tapa cuando se detecte un objeto.
- Los estudiantes explorarán los beneficios ambientales de los sistemas inteligentes de gestión de residuos.

Materiales::

- Placas Arduino Uno (una por estudiante o grupo)
- Sensor ultrasónico HC-SR04 (uno por estudiante o grupo)
- Servomotores (uno por estudiante o grupo)
- Tableros de conexiones y cables puente
- Pequeña caja de cartón o contenedor (para el basurero)
- Tapa de cartón o plástico (para simular la tapa del basurero)
- Cables USB para la programación
- Computadoras con el IDE de Arduino instalado
- Proyector o pizarra para las demostraciones

Actividades:

Día1

Introducción a Arduino y Electrónica (15 minutos):

- Comienza la lección presentando Arduino como una plataforma de microcontrolador utilizada para diversos proyectos.
- Discute la importancia de la electrónica y la automatización en la tecnología moderna.

Sensores Ultrasónicos y Detección de Objetos (30 minutos):

- Explica el principio de los sensores ultrasónicos y cómo funcionan para la medición de distancias.
- Habla sobre los componentes del sensor ultrasónico HC-SR04 (transmisor y receptor).
- Ilustra cómo se pueden utilizar los sensores ultrasónicos para detectar objetos y medir distancias.

Introducción a los Servomotores (20 minutos):

- Presenta los servomotores y sus aplicaciones en el control de movimientos mecánicos.
- Muestra ejemplos de cómo se pueden utilizar los servomotores en proyectos, como abrir una tapa.

Preparación de la Actividad Práctica (45 minutos):

- Proporciona a cada grupo un Arduino Uno, un sensor ultrasónico, un servomotor, una placa de conexiones y cables puente.
- Instruye a los estudiantes para que ensamblen el prototipo del Smart Dustbin, posicionando adecuadamente el sensor ultrasónico y el servomotor.

Día2

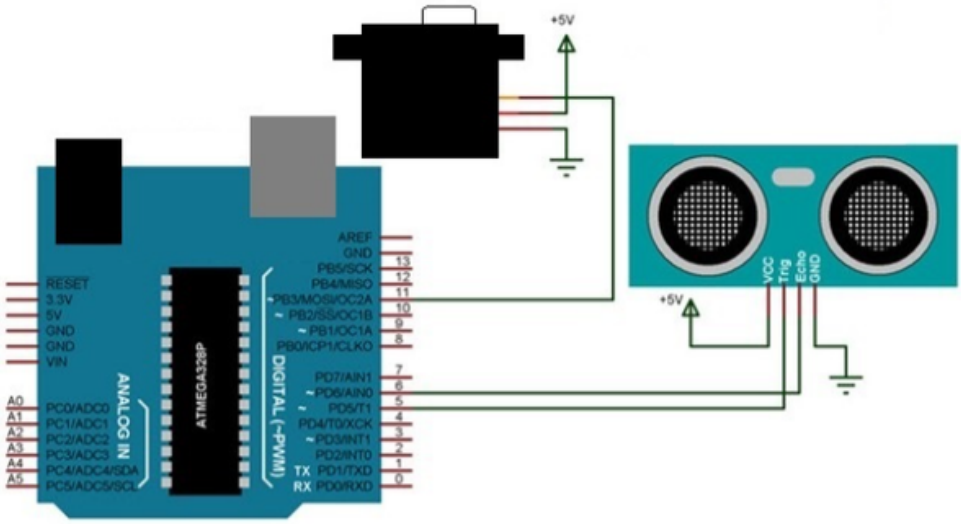
Fundamentos de la Programación en Arduino (30 minutos):

- Enseña a los estudiantes los fundamentos de la programación en Arduino, incluyendo setup(), loop(), y pinMode().
- Proporciona ejemplos de código básico de Arduino para hacer parpadear un LED.

Programación del Smart Dustbin (60 minutos):

- Guía a los estudiantes en la escritura de código Arduino para controlar el Smart Dustbin basado en las lecturas del sensor ultrasónico.
- Explica la lógica para abrir la tapa cuando se detecta un objeto dentro de un rango específico.

shemá:



Aquí tienes un ejemplo de código Arduino para un Smart Dustbin que utiliza un sensor ultrasónico (HC-SR04) y un servomotor. Este código permite que el servomotor abra la tapa del basurero cuando se detecta un objeto dentro de un rango específico:

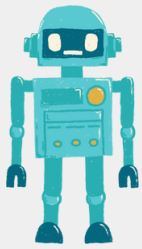
```
#include <Servo.h>

#define TRIGGER_PIN 9
#define ECHO_PIN 10
#define SERVO_PIN 11

Servo myservo; // Create a Servo object
int distance; // Variable to store distance measured by the Ultrasonic sensor

void setup() {
 myservo.attach(SERVO_PIN); // Attach the Servo to the specified pin
 pinMode(TRIGGER_PIN, OUTPUT);
 pinMode(ECHO_PIN, INPUT);
 Serial.begin(9600); // Initialize serial communication for debugging
}
```

```
void loop() {
 // Send a brief pulse to trigger the Ultrasonic sensor
 digitalWrite(TRIGGER_PIN, LOW);
 delayMicroseconds(2);
 digitalWrite(TRIGGER_PIN, HIGH);
 delayMicroseconds(10);
 digitalWrite(TRIGGER_PIN, LOW);
 // Read the duration of the echo pulse and calculate the distance
 duration = pulseIn(ECHO_PIN, HIGH);
 distance = (duration / 2) / 29.1; // Calculate distance in centimeters
 // Check if an object is within the specified range (adjust as needed)
 if (distance < 20) { // You can adjust the distance threshold here
  // If an object is detected, open the flap
   myservo.write(90); // Rotate the Servo to open the flap (adjust the angle as needed)
  delay(1000); // Wait for 1 second
  myservo.write(0); // Rotate the Servo back to close the flap
 }
 // Print the distance to the Serial Monitor for debugging
 Serial.print("Distance: ");
 Serial.print(distance);
 Serial.println(" cm");
 // Add a delay between readings to prevent rapid triggering
 delay(1000); // You can adjust the delay time as needed
}
```



En este código:

- Incluimos la biblioteca Servo y definimos los números de pin para el trigger y echo del sensor ultrasónico, así como el pin de control del servomotor.
- En la función setup(), adjuntamos el servomotor al pin especificado y configuramos el pin trigger como salida y el pin echo como entrada. También inicializamos la comunicación serial para fines de depuración.
- La función loop() realiza repetidamente los siguientes pasos:
  1. Envía un breve pulso al sensor ultrasónico para activar una medición de distancia.
  2. Mide la duración del pulso de eco y calcula la distancia en centímetros.
  3. Comprueba si un objeto está dentro del rango especificado (20 cm en este ejemplo) y, si es así, abre la tapa del basurero girando el servomotor a un ángulo especificado (90 grados).
  4. Imprime la distancia en el Monitor Serial con fines de depuración.
  5. Agrega un retraso entre las lecturas para evitar activaciones rápidas.

Puedes ajustar el umbral de distancia, el ángulo del servomotor y los tiempos de retraso para que se adapten a tu configuración específica y tus requisitos.

Pruebas y Solución de Problemas (30 minutos):

- Anima a los estudiantes a probar sus prototipos de Smart Dustbin y a solucionar cualquier problema con el código o el hardware.
- Fomenta la experimentación con diferentes distancias de detección y ángulos del servomotor.

Día3

- Presentación del Proyecto y Discusión (60 minutos):
- Cada grupo presenta su proyecto de Smart Dustbin a la clase, explicando el diseño, los componentes y el código.
- Discute los beneficios ambientales de los sistemas inteligentes de gestión de residuos y su impacto potencial en la reducción de residuos.

Discusión Abierta y Futuras Mejoras (30 minutos):

- Facilita una discusión en clase sobre posibles mejoras en el Smart Dustbin y otras aplicaciones de tecnología similar.
- Anima a los estudiantes a hacer lluvia de ideas sobre ideas para mejorar aún más las soluciones de gestión de residuos.

Evaluaciones

Evaluar a los estudiantes en función de su participación, la funcionalidad del proyecto, su comprensión de la programación en Arduino y su capacidad para solucionar problemas.

Tarea

Asignar tarea que requiere que los estudiantes investiguen y presenten ejemplos reales de sistemas de gestión de residuos inteligentes y su impacto en la sostenibilidad.

Conclusión:

Concluye la lección resumiendo los conceptos clave aprendidos, destacando la intersección entre la tecnología y las soluciones ambientales, y alentando a los estudiantes a pensar críticamente en la aplicación de la tecnología para generar cambios positivos.

FROM GAME TO STEM

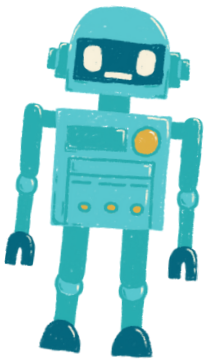
Co-funded by the Erasmus+ Programme of the European Union

Lección 10
Lección de Ecología con Arduino

Explorando la Calidad del Aire con Arduino

Duración:

1 período



Objetivo:

- Los estudiantes aprenderán sobre la importancia de la calidad del aire en la salud ambiental.
- Los estudiantes comprenderán cómo los contaminantes del aire pueden afectar a los ecosistemas y la salud humana.
- Los estudiantes construirán un sensor de calidad del aire simple basado en Arduino y recopilarán datos.
- Los estudiantes discutirán la importancia de monitorear la calidad del aire para el bienestar ecológico.

Materiales:

- Placas Arduino Uno (una por estudiante o grupo)
- Módulo de sensor de calidad del aire (por ejemplo, MQ-135)
- Cables puente
- Cables USB para la programación
- Computadoras con Arduino IDE instalado
- Proyector o pizarra para las demostraciones

Actividades

Día1

Introducción a la Calidad del Aire (10 minutos):

- Comienza la lección discutiendo la importancia del aire limpio y su impacto en los ecosistemas.
- Explica cómo los contaminantes del aire pueden afectar tanto a los entornos naturales como a la salud humana.

Sensores de Calidad del Aire (20 minutos):

- Introduce los sensores de calidad del aire y su papel en la monitorización de la contaminación del aire.
- Explica el funcionamiento básico de los sensores de calidad del aire y cómo miden varios gases.

Conceptos Básicos de Arduino (10 minutos):

- Presenta Arduino como una plataforma de microcontrolador para la recopilación de datos.
- Muestra a los estudiantes los componentes de una placa Arduino y los conceptos básicos de la escritura y carga de código.

Preparación de la Actividad Práctica (15 minutos):

- Proporciona a cada grupo una placa Arduino Uno, un módulo de sensor de calidad del aire y cables puente.
- Instruye a los estudiantes para que ensamblen el sensor de calidad del aire y lo conecten a Arduino.

Construcción y Programación del Sensor de Calidad del Aire (20 minutos):

- Guía a los estudiantes en la construcción de su sistema de sensor de calidad del aire basado en Arduino.
- Muestra a los estudiantes cómo escribir código Arduino para recopilar datos de calidad del aire del sensor.

Aquí tienes un ejemplo sencillo de código Arduino para monitorear la calidad del aire utilizando un sensor de calidad del aire MQ-135. Este código lee los datos del sensor y los muestra en el Monitor Serie. Asegúrate de tener las bibliotecas adecuadas instaladas en tu Arduino IDE, ya que el sensor MQ-135 puede requerir calibración para obtener resultados precisos.

```
// Include necessary libraries
#include <Wire.h>
#include <Adafruit_Sensor.h>
#include <Adafruit_MQ135.h>

// Define the analog pin where the MQ-135 sensor is connected
#define MQ135_PIN A0

// Create an instance of the Adafruit MQ135 class
Adafruit_MQ135 mq135(MQ135_PIN);

void setup() {
 // Initialize serial communication for data output
 Serial.begin(9600);
}

void loop() {
 // Read the MQ-135 sensor's data
 float airQuality = mq135.readCO2();

 // Print the air quality data to the Serial Monitor
 Serial.print("Air Quality: ");
 Serial.print(airQuality);
 Serial.println(" ppm");

 // Add a delay before the next reading
 delay(2000); // Adjust the delay time as needed
}
```

En este código:

- Incluimos las bibliotecas necesarias, incluyendo las bibliotecas Adafruit Sensor y Adafruit MQ135, que son comúnmente utilizadas para trabajar con el sensor de calidad del aire MQ-135. Asegúrate de tener estas bibliotecas instaladas en tu entorno de desarrollo de Arduino.
- Definimos el pin analógico (A0 en este ejemplo) al cual está conectado el sensor MQ-135 en Arduino.
- Creamos una instancia de la clase Adafruit_MQ135 para interactuar con el sensor.
- En la función setup(), inicializamos la comunicación serial para la salida de datos en el Monitor Serie.
- En la función loop(), leemos continuamente los datos de calidad del aire del sensor MQ-135 utilizando mq135.readCO2(). Los datos representan la concentración de CO2 en partes por millón (ppm).
- Los datos de calidad del aire se imprimen en el Monitor Serie, y hay un retraso de 2 segundos (ajustable) antes de tomar la próxima lectura.

Ten en cuenta que el sensor MQ-135 puede requerir calibración para obtener lecturas precisas, y el código proporcionado aquí sirve como un ejemplo básico. Dependiendo de tu aplicación específica y los requisitos de calibración, es posible que necesites ajustar el código y el proceso de calibración en consecuencia.

Recopilación de Datos y Discusión (20 minutos):

- Instruye a los estudiantes a recopilar datos de calidad del aire ejecutando sus sensores en diferentes entornos (por ejemplo, en interiores, cerca de una carretera, en un jardín).
- Pide a los estudiantes que registren y compartan sus datos con la clase.
- Lidera una discusión en clase sobre las diferencias en los datos de calidad del aire y las posibles consecuencias ecológicas.

Evaluaciones

Evalúa a los estudiantes en función de su participación, la recopilación de datos y su capacidad para discutir el impacto de la calidad del aire en los sistemas ecológicos.

Terea

Asigna tarea que requiera que los estudiantes investiguen y presenten ejemplos del mundo real de problemas ecológicos relacionados con la contaminación del aire y la importancia de la monitorización de la calidad del aire para mitigar estos problemas.

Conclusión:

Para concluir la lección, es importante resumir los conceptos ecológicos clave aprendidos y enfatizar el papel de la tecnología, en este caso Arduino, en la monitorización ambiental.

FROM GAME TO STEM

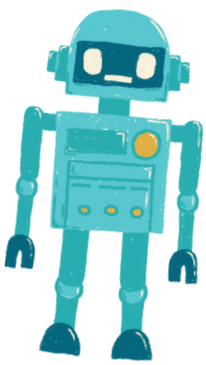
Co-funded by the Erasmus+ Programme of the European Union

Lekcija 1
Nastava matematike uz Arduino

Trigonometrija uz Arduino

Trajanje:

3 perioda



Cilj:

- Učenici će razumeti osnovne trigonometrijske pojmove, uključujući sinus, kosinus i tangens.
- Učenici će primeniti trigonometrijske funkcije kako bi rešili stvarne probleme.
- Učenici će programirati Arduino uređaj kako bi napravili jednostavan uređaj za merenje ugla koristeći servo motor.

Materijal:

- Arduino ploče (jedna po učeniku ili grupi)
- Servo motori
- Breadboard-ovi
- Spojnice
- Šestar
- Lenjir
- USB kablovi za programiranje
- Računari sa instaliranim Arduino IDE
- Projektor ili bela tabla za demonstracije

Aktivnosti

Dan 1

Uvod u trigonometriju (15 minuta):

- Započnite lekciju uvodeći pojam trigonometrije i njenu važnost u stvarnim primenama.
- Kratko objasnite sinus, kosinus i tangens kao odnose stranica u pravouglim trouglovima.
- Razgovarajte o tome kako se trigonometrijske funkcije mogu koristiti za rešavanje problema koji uključuju uglove i razdaljine.

Osnove trigonometrije (30 minuta):

- Dublje se upustite u definicije sinusa, kosinusa i tangensa.
- Dajte primere kako se ove funkcije koriste za pronalaženje nedostajućih uglova ili dužina stranica u pravouglim trouglovima.
- Dopustite učenicima da vežbaju rešavanje osnovnih trigonometrijskih problema na papiru.

Uvod u arduino i servo motore (20 minuta):

- Predstavite Arduino kao platformu za kreiranje uređaja i objasnite njegove komponente (mikrokontroler, senzori, aktuatori).
- Objasnite pojam servo motora i kako ih možete kontrolisati da se kreću na određene uglove.
- Pokažite primere upravljanja servo motorima pomoću Arduino-a.

Praktična aktivnost (45 minuta):

- Podelite učenike u parove ili male grupe.
- Dajte svakoj grupi Arduino ploču, servo motor, breadboard i spojnice.
- Naučite učenike kako da sastave jednostavan uređaj za merenje ugla pomoću servo motora i upotrebom protraktora kao referencom.
- Vodite učenike u pisanju Arduino koda za kontrolu servo motora kako bi se pomerali na određeni ugao na osnovu korisničkog unosa

Dan 2

Pregled koncepata trigonometrije (20 minuta):

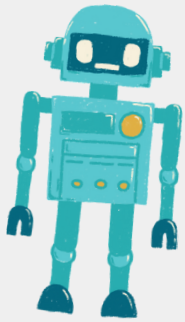
- Započnite drugi dan pregledom pojmova sinusa, kosinusa i tangensa naučenih prvog dana.
- Dajte dodatne probleme za vežbu koje učenici treba da reše koristeći trigonometrijske funkcije.

Programiranje uređaja za merenje ugla (45 minuta):

- Nastavite s praktičnom aktivnošću započetom prvog dana.
- Naredite učenicima da završe svoj Arduino kod kako bi tačno merili i prikazivali uglove pomoću servo motora i numeričkog prikaza (npr. Serial Monitor).
- Podstaknite učenike da primene trigonometriju kako bi pretvorili poziciju serva u uglove."

Evo primera Arduino koda za kreiranje jednostavnog uređaja za merenje ugla pomoću servo motora. Ovaj kod omogućava servo motoru da se pomeri na određeni ugao na osnovu unosa korisnika i prikazuje izmereni ugao na serijskom monitoru:

```
#include <Servo.h>
Servo myservo; // Create a Servo object
void setup() {
myservo.attach(9); // Attaches the servo to digital pin 9
Serial.begin(9600); // Initialize serial communication for debugging
}
void loop() {
int angle; // Variable to store the desired angle
Serial.println("Enter an angle (0-180): ");
while (!Serial.available()) {
  // Wait for user input
}
angle = Serial.parseInt(); // Read the angle entered by the user
if (angle >= 0 && angle <= 180) {
// Check if the entered angle is within the valid range
myservo.write(angle); // Move the servo to the specified angle
delay(500); // Delay for stability
Serial.print("Measured angle: ");
Serial.print(angle);
Serial.println(" degrees");
} else {
Serial.println("Invalid angle. Please enter a value between 0 and 180.");
}
}
```



U ovom kodu:

Uključujemo Servo biblioteku za kontrolu servo motora.
U funkciji setup(), priključujemo servo na digitalni pin 9 i inicijalizujemo serijsku komunikaciju za otklanjanje grešaka.
U funkciji loop() čitamo unos korisnika za željeni ugao koristeći serijski monitor. Od korisnika se traži da unese ugao između 0 i 180 stepeni.
Proveravamo da li je uneti ugao unutar važećeg opsega (0 do 180 stepeni). Ako jeste, pomeramo servo na određeni ugao koristeći miservo.vrite(angle) i prikazujemo izmereni ugao na serijskom monitoru.
Ako je uneti ugao izvan važećeg opsega, prikazujemo poruku o grešci.
Da biste koristili ovaj kod sa svojim učenicima, uverite se da su spojili servo motor na digitalni pin 9 na Arduinu i otvorili serijski monitor da unesu uglove i posmatraju izmerene uglove.

Dan 3

Prezentacija projekta i diskusija (60 minuta):

- Svaka grupa prezentuje svoj uređaj za merenje ugla pomoću Arduino-a pred celim razredom.
- Ohrabrite učenike da objasne kako su primenili trigonometrijske koncepte u svojim projektima.
- Razgovarajte o stvarnim primenama uređaja za merenje ugla i trigonometrije.
- Olakšajte diskusiju u razredu o izazovima i rešenjima koje su susreli tokom projekta."

Procena

Ocenjujte učenike na osnovu njihovog učešća, kvaliteta koda, tačnosti merenja uglova i njihove sposobnosti da objasne trigonometrijske principe iza svog uređaja.

Domaći

Dodelite domaći zadatak koji zahteva od učenika da istraže i prezentuju stvarnu primenu trigonometrije u različitim oblastima, kao što su inženjering, fizika ili arhitektura.

Zaključak

Zaključite čas sažimanjem ključnih trigonometrijskih koncepata naučenih tokom časa i istaknite njihovu praktičnu primenu u razvoju Arduino uređaja. Naglasite vezu između matematike i tehnologije.

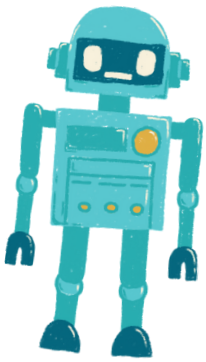
FROM GAME TO STEM

Co-funded by the
Erasmus+ Programme
of the European Union

Lekcija 2
Nastava matematike uz Arduino
Algebarsko Modeliranje sa Arduinom

Trajanje:

2 perioda



Cilj:

- Učenici će naučiti kako koristiti algebarske koncepte za rešavanje stvarnih problema.
- Učenici će primeniti matematičko modeliranje i veštine programiranja kako bi kreirali Arduino uređaj koji rešava praktičan problem.
- Učenici će sticati iskustvo u radu sa varijablama i jednačinama.

Materijal:

- Arduino ploče (jedna po učeniku ili grupi)
- Breadboard-ovi
- Senzori ili ulazni uređaji (npr. temperaturni senzor, svetlo senzor, taster)
- LED diode, otpornici i spojnice
- USB kablovi za programiranje
- Računari sa instaliranim Arduino IDE
- Projektor ili bela tabla za demonstracije

Aktivnosti

Dan 1

<u>Uvod u algebarsko modeliranje (15 minuta):</u>

- Započnite čas uvodom u algebarsko modeliranje i njegovu relevantnost u rešavanju stvarnih problema.
- Diskutujte o važnosti varijabli i jednačina u matematičkom modeliranju.

<u>Varijable i jednačine (30 minuta):</u>

- Pregledajte koncept varijabli i kako se koriste za predstavljanje nepoznatih vrednosti.
- Uvedite linearnu jednačinu (npr. y=mx+b) kao način modeliranja odnosa između varijabli.
- Dajte primere stvarnih problema koji se mogu predstaviti upotrebom jednačina.

<u>Osnove arduina (20 minuta):</u>

Predstavite Arduino kao platformu za kreiranje uređaja i objasnite njegove komponente. Prikažite primere jednostavnih Arduino projekata i kako se varijable mogu koristiti u programiranju.

<u>Praktična aktivnost (45 minuta):</u>

- Podelite učenike u parove ili male grupe.
- Dajte svakoj grupi Arduino ploču, breadboard, senzor ili ulazni uređaj (npr. temperaturni senzor) i LED.
- Naredite učenike da kreiraju jednostavan Arduino projekt koji koristi senzor za čitanje podataka i LED diodu za vizualno predstavljanje podataka.
- Ohrabrite učenike da koriste varijable za čuvanje očitavanja senzora i kreiranje jednačina za kontrolu LED diode na osnovu podataka sa senzora.
- Diskutujte različite stvarne scenarije u kojima bi njihov uređaj mogao biti koristan.

Dan 2

Pregled algebarskih koncepata (20 minuta):

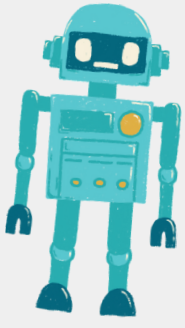
- Započnite drugi dan pregledom koncepata varijabli, jednačina i matematičkog modeliranja.
- Diskutujte o projektima Arduino uređaja s prethodnog dana i njihovim primenama.

Programiranje Algebarskog Modela (45 minuta):

- Nastavite s praktičnom aktivnošću započetom prvog dana.
- Naredite učenicima da modifikuju svoj Arduino kod kako bi kreirali matematički model koristeći podatke senzora.
- Ohrabrite ih da eksperimentišu sa različitim jednačinama i varijablama kako bi predstavili odnos između podataka sa senzora i izlaza na LED diodi.
- Diskutujte kako se model može koristiti za pravljenje predviđanja ili kontrolu sistema u stvarnom svetu.

Evo primera Arduino koda za jednostavan projekat algebarskog modeliranja. U ovom projektu, učenici će koristiti temperaturni senzor za čitanje podataka o temperaturi i kontrolisati LED diodu na osnovu očitavanja temperature. Kod koristi varijable i jednačinu da odredi kada će se LED dioda uključiti ili isključiti.

```
// Define the pins for the temperature sensor, LED, and a resistor (if needed)
const int temperatureSensorPin = A0; // Analog pin for the temperature sensor
const int ledPin = 13; // Use the built-in LED on most Arduino boards
const float thresholdTemperature = 25.0; // Define the threshold temperature
void setup() {
pinMode(ledPin, OUTPUT); // Set the LED pin as an output
Serial.begin(9600); // Initialize serial communication for debugging (optional)
}
void loop() {
// Read the temperature from the sensor
int sensorValue = analogRead(temperatureSensorPin);
// Convert the analog value to temperature in degrees Celsius
float temperatureCelsius = map(sensorValue, 0, 1023, 0, 100); // Adjust the mapping as needed
// Check if the temperature is above the threshold
if (temperatureCelsius > thresholdTemperature) {
digitalWrite(ledPin, HIGH); // Turn the LED on
} else {
digitalWrite(ledPin, LOW); // Turn the LED off
}
// Print the temperature to the serial monitor for debugging (optional)
Serial.print("Temperature: ");
Serial.print(temperatureCelsius);
Serial.println(" °C");
// Delay for a moment to avoid rapid LED toggling
delay(1000); // Delay for 1 second
}
```



U ovom kodu:

Definišemo pinove za temperaturni senzor (povezan na analogni pin), LED diodu (obično ugrađena LED na većini Arduino ploča) i prag temperature koji određuje kada će se LED dioda uključiti.
U funkciji `setup()`, postavljamo pin za LED kao izlaz i inicijalizujemo serijsku komunikaciju za debagovanje (opciono).
U funkciji `loop()`, čitamo temperaturu sa senzora koristeći `analogRead()`. Konvertujemo analognu vrednost senzora u temperaturu u stepenima Celzijusa na osnovu karakteristika senzora.
Zatim proveravamo da li temperatura prelazi definisani prag (thresholdTemperature). Ako je temperatura iznad praga, LED se uključuje; inače se isključuje.
Opciono: Prikazujemo temperaturu na serijskom monitoru u svrhu debagovanja.
Da biste koristili ovaj kod, učenici će morati da povežu temperaturni senzor (npr. LM35) sa analognim pinom na Arduino ploči i LED diodu sa digitalnim pinom. Kod će čitati temperaturu sa senzora i kontrolisati LED diodu na osnovu očitavanja temperature i vrednosti praga.

Prezentacija Projekta i Diskusija (30 minuta):

- Svaka grupa prezentuje svoj Arduino uređaj pred celim razredom.
- Ohrabrite učenike da objasne matematički model koji su kreirali i kako funkcioniše.
- Olakšajte diskusiju u razredu o primenama algebarskog modeliranja u rešavanju praktičnih problema."

Procena

Ocenjujte učenike na osnovu njihovog učešća, funkcionalnosti njihovog Arduino uređaja i njihove sposobnosti da objasne algebarske koncepte koji su korišćeni u njihovim projektima.

Domaći

Dodelite domaći zadatak učenicima koji zahteva istraživanje i prezentaciju stvarnog problema koji se može rešiti primenom algebarskog modeliranja i Arduino uređaja.

Zaključak

Zaključite čas sažimanjem ključnih algebarskih koncepata naučenih tokom časa i naglašavanjem važnosti matematičkog modeliranja u tehnologiji i inženjeringu.

FROM GAME TO STEM

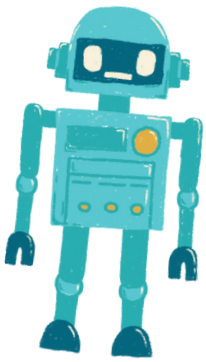
Co-funded by the Erasmus+ Programme of the European Union

Lekcija 3
Nastava fizike sa Arduinom

Istraživanje kretanja i akceleracije sa Arduinom

Trajanje:

3 perioda



Cilj:

- Učenici će razumeti osnovne principe kretanja i akceleracije.
- Učenici će primeniti kinematičke jednačine za rešavanje stvarnih problema koji se odnose na kretanje.
- Učenici će programirati Arduino uređaj kako bi merili i prikazivali akceleraciju pomoću senzora.

Materijal:

- Arduino ploče (jedna po učeniku ili grupi)
- Moduli akcelerometara (na primer, ADXL345)
- Breadboard-ovi
- Spojnice (jumper wires)
- USB kablovi za programiranje
- Računari sa instaliranim Arduino IDE
- Projektor ili bela tabla za demonstracije
- Opciono: Stvarni objekti za eksperimente sa kretanjem (npr. igračke automobile, loptice)

Aktivnosti

Dan 1

<u>Uvod u Kretanje i Akceleraciju (15 minuta):</u>

- Započnite čas uvodom u koncept kretanja i akceleracije.
- Definišite ključne termine poput brzine, akceleracije i usporavanja.
- Diskutujte o važnosti razumevanja kretanja u različitim oblastima kao što su fizika, inženjering i transport.

<u>Fizika Kretanja (30 minuta):</u>

- Objasnite jednačine kretanja, uključujući:
- Pomak (displacement): $s = ut + \frac{1}{2}at^2$
- Brzina (velocity): $v = u + at$
- Akceleracija (acceleration): $a = \frac{v-u}{t}$
- Priložite primere kako se ove jednačine koriste za analizu kretanja.
- Sprovedite jednostavne eksperimente sa kretanjem (npr. kotrljanje loptice niz kosinu) kako biste demonstrirali principe kretanja.

<u>Uvod u Arduino i Akcelerometre (20 minuta):</u>

- Predstavite Arduino kao platformu za kreiranje uređaja i objasnite njegove komponente.
- Objasnite koncept akcelerometara i kako mere akceleraciju.
- Prikažite primere izlaznih podataka akcelerometra i objasnite kako se odnose na stvarno kretanje.

<u>Praktična Aktivnost (45 minuta):</u>

- Podelite učenike u parove ili manje grupe.
- Dajte svakoj grupi Arduino ploču, modul akcelerometra, breadboard i spojnice (jumper wires).
- Naredite učenike da sastave jednostavan sklop za povezivanje akcelerometra sa Arduino pločom.
- Vodite učenike u pisanju Arduino koda za čitanje i prikazivanje podataka o akceleraciji sa senzora na serijskom monitoru.

Dan 2

Pregled Kinematičkih Jednačina (20 minuta):

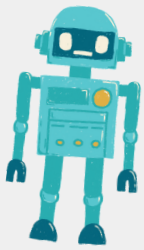
- Započnite drugi dan pregledom kinematičkih jednačina koje su bile predmet diskusij prvog dana.
- Dajte dodatne primere za učenike kako bi rešili koristeći ove jednačine.

Programiranje Uređaja za Merenje Akceleracije (45 minuta):

- Nastavite sa praktičnom aktivnošću započetom prvog dana.
- Naredite učenicima da kompletiraju svoj Arduino kod kako bi čitali i prikazivali podatke o ubrzanju u realnom vremenu sa akcelerometra.
- Ohrabrite učenike da kalibrišu senzor ako je potrebno i primene jednačine za akceleraciju kako bi interpretirali podatke.

Evo primera Arduino koda za merenje i prikazivanje akceleracije pomoću akcelerometra ADXL345. Ovaj kod će omogućiti vašim učenicima da povežu akcelerometar sa Arduino pločom i prikažu vrednosti ubrzanja na serijskom monitoru.

```
#include <Wire.h>
#include <Adafruit_Sensor.h>
#include <Adafruit_ADXL345_U.h>
Adafruit_ADXL345_Unified accel = Adafruit_ADXL345_Unified(12345);
void setup(void) {
 Serial.begin(9600);
 Serial.println("Accelerometer Test"); Serial.println("");
  if(!accel.begin())
 {
/* There was a problem detecting the ADXL345 ... check your connections*/
   Serial.println("Ooops, no ADXL345 detected ... Check your wiring!");
   while(1);
 }
}
void loop(void) {
 sensors_event_t event;
 accel.getEvent(&event);
 /* Display the acceleration values (in m/s^2) */
 Serial.print("X: "); Serial.print(event.acceleration.x); Serial.print(" ");
 Serial.print("Y: "); Serial.print(event.acceleration.y); Serial.print(" ");
 Serial.print("Z: "); Serial.print(event.acceleration.z); Serial.println(" m/s^2");
 delay(500); // Delay for half a second between readings
}
```



U ovom kodu:

Uključujemo potrebne biblioteke za senzor akcelerometra ADXL345.
Kreiramo objekat Adafruit_ADXL345_Unified pod nazivom "accel" kako bismo mogli da komuniciramo sa senzorom.
U funkciji "setup()", inicijalizujemo serijsku komunikaciju za debagiranje i proveravamo da li je senzor akcelerometra detektovan. Ako nije, prikazaće se poruka o grešci.
U funkciji "loop()", kontinuirano čitamo podatke o ubrzanju sa senzora pomoću funkcije "accel.getEvent(&event)" i prikazujemo vrednosti ubrzanja po X, Y i Z osama u metrima kvadratnim u sekundi (m/s^2) na serijskom monitoru.
Da biste koristili ovaj kod, pobrinite se da vaši učenici ispravno povežu senzor akcelerometra ADXL345 sa Arduino pločom. Senzor treba da bude povezan sa odgovarajućim pinovima (na primer, SDA i SCL) za I2C komunikaciju. Kada se Arduino uključi i pokrene ovaj kod, kontinuirano će prikazivati podatke o ubrzanju na serijskom monitoru.
Napomena: Moguće je da će biti potrebno instalirati Adafruit ADXL345 biblioteku putem Arduino Library Manager-a kako bi se koristio ovaj kod.

Prezentacija Projekta i Diskusija (60 minuta):

- Svaka grupa prezentuje svoj Arduino uređaj za merenje ubrzanja pred celim razredom.
- Ohrabrite učenike da objasne kako su primenili kinematičke jednačine i principe fizike u svojim projektima.
- Diskutujte o stvarnim primenama uređaja za merenje ubrzanja u različitim oblastima.
- Olakšajte diskusiju u razredu o izazovima i rešenjima sa kojima su se suočili tokom projekta.

Procena

Ocenjujte učenike na osnovu njihovog učešća, kvaliteta koda, tačnosti merenja ubrzanja i njihove sposobnosti da objasne fizikalne principe iza svog uređaja.

Domaći

Dodelite domaći zadatak učenicima koji zahteva istraživanje i prezentaciju stvarne primene merenja ubrzanja u fizici ili inženjeringu.

Zaključak

Zaključite čas sažimanjem ključnih fizikalnih koncepata naučenih i naglašavanjem njihove praktične primene u razvoju Arduino uređaja. Istaknite vezu između fizike i tehnologije.

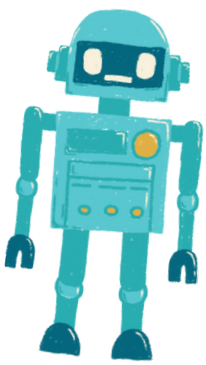
Co-funded by the
Erasmus+ Programme
of the European Union

Lekcija 4
Nastava fizike sa Arduinom

Istraživanje oscilacija i kretanja klatna sa Arduinom

Trajanje:

2 perioda



Cilj:

- Učenici će razumeti principe oscilacionog kretanja i harmoničnih oscilacija.
- Učenici će primeniti matematičko modeliranje kako bi analizirali kretanje klatna.
- Učenici će programirati Arduino kako bi simulirali i vizualizovali kretanje klatna.

Materijal:

- Arduino ploče (jedna po učeniku ili grupi)
- Servo motori
- Breadboard-ovi
- Spojnice (jumper wires)
- Mali objekti kao tegovi za klatno (npr. podloške)
- Konac ili nit za klatno
- Lenjir ili merač trake
- USB kablovi za programiranje
- Računari sa instaliranim Arduino IDE
- Projektor ili bela tabla za demonstracije

Aktivnosti

Dan 1

Uvod u Oscilacije i Kretanje Klatna (15 minuta):

- Započnite čas uvodom u koncept oscilatornog kretanja i njegovu važnost u različitim oblastima, uključujući fiziku i inženjering.
- Definišite ključne termine kao što su period, frekvencija, amplituda i harmonične oscilacije.
- Objasnite osnove kretanja penduluma i njegovo matematičko predstavljanje.

Matematičko Modeliranje Klatna (30 minuta):

- Diskutujte matematički model jednostavnog penduluma, uključujući jednačinu za period pe $T = 2\pi\sqrt{\frac{L}{g}}$
- gdje je T period, L je dužina penduluma, a g je ubrzanje usled gravitacije.
- Dajte primere kako koristiti ovu jednačinu za izračunavanje perioda penduluma.
- Sprovedite jednostavne izračune koji se odnose na kretanje penduluma.

Uvod u Arduino i Servo Motore (20 minuta):

- Predstavite Arduino kao platformu za kreiranje uređaja i objasnite njegove komponente (mikrokontroler, senzori, aktuatori).
- Objasnite koncept servo motora i kako se mogu koristiti za simulaciju kretanja penduluma.
- Prikažite primere kontrole servo motora pomoću Arduino-a.

Praktična Aktivnost (45 minuta):

- Podelite učenike u parove ili manje grupe.
- Dajte svakoj grupi Arduino ploču, servo motor, breadboard, spojnice, mali objekat kao teg za pendulum i konopac.
- Naredite učenike da sastave jednostavan simulator penduluma koristeći servo motor za kontrolu kretanja penduluma.
- Vodite učenike u pisanju Arduino koda kako bi stvorili harmonične oscilacije varirajući položaj servo motora.

Day 2

Pregled Kretanja Klatna (20 minuta):

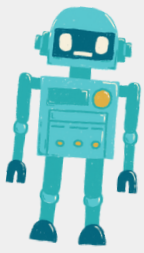
- Započnite drugi dan pregledom koncepata oscilacionog kretanja, kretanja pendulum i matematičkog modela jednostavnog penduluma.
- Diskutujte o projektima sa Arduinom sa prethodnog dana i njihovim primenama.

Programiranje Simulatora klatna (45 minuta):

- Nastavite sa praktičnom aktivnošću započetom prvog dana.
- Naredite učenicima da modifikuju svoj Arduino kod kako bi tačno simulirali kretanje penduluma sa različitim parametrima kao što su dužina i amplituda.
- Ohrabrite učenike da vizualizuju i grafički prikažu kretanje koristeći servo motor.

Evo primera Arduino koda za kreiranje jednostavnog simulatora penduluma pomoću servo motora. Ovaj kod omogućava učenicima da programiraju Arduino kako bi simulirali i vizualizovali kretanje klatna:

```
#include <Servo.h>
Servo pendulum; // Create a Servo object
int angle = 90; // Initial angle of the pendulum (straight down)
int amplitude = 45; // Maximum angle to one side of the vertical (adjust as needed)
int period = 2000; // Period of the pendulum swing in milliseconds (adjust as needed)
void setup() {
  pendulum.attach(9); // Attach the servo to digital pin 9
}
void loop() {
  // Calculate the angle of the pendulum using harmonic motion
  int displacement = amplitude * cos(2 * PI * millis() / period);
  int pendulumAngle = angle + displacement;
   // Move the servo to the calculated angle
  pendulum.write(pendulumAngle);
  // Delay for a short time to control the speed of the simulation
  delay(50); // Adjust as needed for desired speed
}
```



U ovom kodu:
Uključujemo Servo biblioteku kako bismo kontrolisali servo motor.
Kreiramo objekat Servo pod nazivom "pendulum" kako bismo kontrolisali servo motor.
Definišemo promenljive za početni ugao penduluma (angle), maksimalni ugao sa jedne strane vertikale (amplitude) i period kretanja penduluma (period). Možete prilagoditi ove vrednosti kako biste promenili ponašanje penduluma.
U funkciji "setup()", pričvršćujemo servo motor za digitalni pin 9.
U funkciji "loop()", izračunavamo ugao penduluma koristeći formulu za harmonično kretanje. Displasment predstavlja ugao odstupanja od vertikalnog položaja, i dodajemo ga na početni ugao kako bismo dobili "pendulumAngle".
Koristimo "pendulum.write(pendulumAngle)" kako bismo pomerili servo motor na izračunati ugao.
Dodajemo odgodu (delay) kako bismo kontrolisali brzinu simulacije. Prilagodite vrednost odgode prema potrebi kako biste postigli željenu brzinu simulacije.
Da biste koristili ovaj kod, učenici treba da povežu servo motor sa digitalnim pinom 9 na Arduino ploči i da prenesu kod na ploču. Servo motor će simulirati kretanje penduluma, a učenici će moći da posmatraju oscilacije u akciji.

Prezentacija Projekta i Diskusija (30 minuta):

- Svaka grupa prezentuje svoj Arduino simulator penduluma pred celim razredom. Ohrabrite učenike da objasne kako su primenili principe matematičkog modeliranja u svojim projektima. Diskutujte o stvarnim primenama razumevanja harmoničnih oscilacija u oblastima kao što su fizika, inženjering i astronomija. Olakšajte diskusiju u razredu o izazovima i rešenjima sa kojima su se suočili tokom projekta.

Procena

Ocenjujte učenike na osnovu njihovog učešća, kvaliteta njihovog Arduino uređaja i njihove sposobnosti da objasne principe oscilacionog kretanja i harmoničnih oscilacija.

Domaći

Dodelite domaći zadatak učenicima koji zahteva istraživanje i prezentaciju stvarne primene oscilacija i harmoničnog kretanja u nauci ili tehnologiji.

Zaključak

Zaključite čas sažimanjem ključnih fizikalnih koncepata naučenih i naglašavanjem njihove praktične primene u razvoju Arduino uređaja. Istaknite vezu između matematike i tehnologije u razumevanju kretanja penduluma.

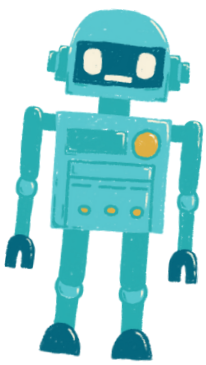
FROM GAME TO STEM

Co-funded by the Erasmus+ Programme of the European Union

Lekcija 5
Nastava hemije sa Arduinom

Istraživanje hidroponike i hemije rasta biljaka

Trajanje:
3 perioda



Objective:

- Učenici će razumeti principe hidroponike i njene prednosti u rastu biljaka.
- Učenici će naučiti o osnovnim hranljivim materijama za rast biljaka i kako pripremiti rastvore hranljivih materijala.
- Učenici će dizajnirati i izgraditi automatizirani sistem za navodnjavanje pomoću Arduino platforme.
- Učenici će pratiti i kontrolisati nivo pH vrednosti u hidroponskom sistemu kako bi optimizovali rast biljaka.

Materials:

- Arduino ploče (jedna po učeniku ili grupi)
- pH senzori i kalibracioni rastvori za pH
- Vodene pumpe
- Crevo i kapanje mlaznice (drip emitters)
- Rezervoar za rastvor hranljivih materijala
- Rastvori za podešavanje pH vrednosti (pH up i pH down)
- Kalibracioni rastvori za pH
- Semenke ili mladice biljaka
- Hidroponski medijum za rast (npr. kamena vuna, perlit)
- Komponente za rastvor hranljivih materijala (npr. N-P-K đubrivo)
- Kontejneri za hidroponske postavke
- Spojnice (jumper wires)
- USB kablovi za programiranje
- Računari sa instaliranim Arduino IDE
- Projektor ili bela tabla za demonstracije

Aktivnosti

Dan 1

Introduction to Hydroponics (15 minutes):

- Započnite čas uvodom u koncept hidroponike i njene prednosti u rastu biljaka.
- Diskutujte o prednostima kontrolisanih okruženja i sistema za efikasno korišćenje vode.

Hemija Rasta Biljaka (30 minuta):

- Objasnite hemijske elemente koji su neophodni za rast biljaka (npr. azot, fosfor, kalijum) i njihove uloge.
- Diskutujte o važnosti nivoa pH vrednosti u unosu hranljivih materijala i zdravlju biljaka.
- Uvedite koncept rastvora hranljivih materijala i kontrole pH vrednosti u hidroponici.

Uvod u Arduino i Senzore (20 minuta):

- Predstavite Arduino kao platformu za automatizaciju i prikupljanje podataka i objasnite njegove komponente (mikrokontroler, senzori).
- Prikažite primere senzora koji se koriste u hidroponskim sistemima (npr. pH senzori).

Priprema za Praktičnu Aktivnost (45 minuta):

- Dajte učenicima Arduino ploče, pH senzore, vodene pumpe, creva i kontejnere.
- Naredite učenicima da razmisle i planiraju dizajn svog hidroponskog sistema, uključujući rastvore hranljivih materijala i kontrolu pH vrednosti.

Dan 2

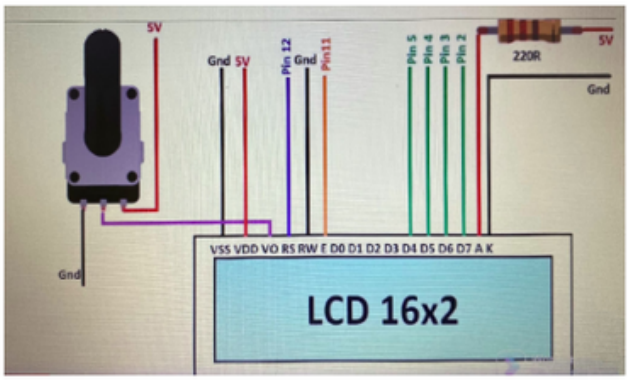
Postavljanje Hidroponskog Sistema (60 minuta):

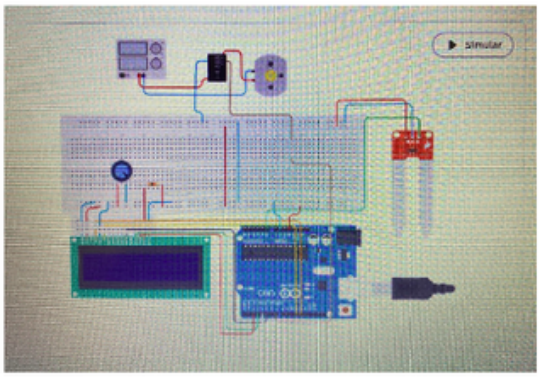
- Započnite drugi dan omogućavanjem učenicima da postave svoje hidroponske sisteme.
- Učenici treba da zasade semenke ili mladice biljaka u hidroponski rastući medijum (npr. kamena vuna) i organizuju sistem za navodnjavanje koristeći creva i kapanje mlaznice (drip emitters).
- Demonstrirajte kako se mešaju i pripremaju rastvori hranljivih materijala.

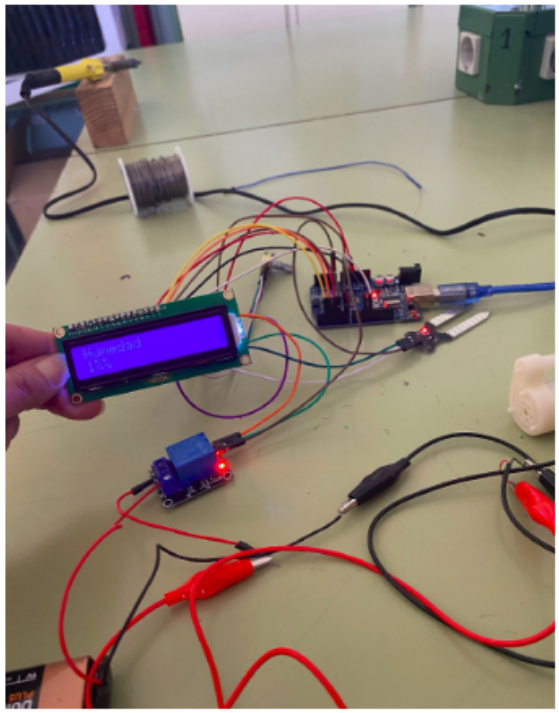
Pratnja i Kalibracija pH Vrednosti (30 minuta):

- Naredite učenicima kako da kalibrišu i koriste pH senzore za praćenje pH vrednosti.
- Objasnite važnost održavanja pH vrednosti unutar optimalnog opsega za unos hranljivih materijala (obično oko pH 6-7).
- Vodite učenike u kalibraciji njihovih pH senzora koristeći rastvore kalibracije pH vrednosti.

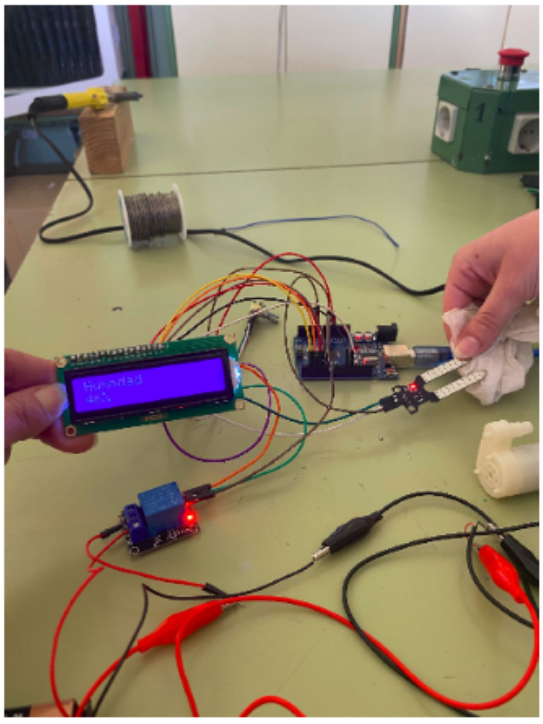
Šema







1% vlažnosti: pumpa radi



46% vlažnosti: pumpa i dalje radi

Dan 3

Postavljanje Hidroponskog Sistema (60 minuta):

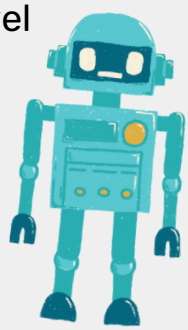
- Započnite drugi dan tako što ćete omogućiti učenicima da postave svoje hidroponske sisteme.
- Učenici treba da zasade semenke ili mladice biljaka u hidroponski rastući medijum (npr. kamena vuna) i da organizuju sistem za navodnjavanje koristeći creva i kapanje mlaznice (drip emitters).
- Demonstrirajte kako se mešaju i pripremaju rastvori hranljivih materijala.

Pratnja i Kalibracija pH Vrednosti (30 minuta):

- Instruirajte učenike kako da kalibrišu i koriste pH senzore za praćenje pH vrednosti.
- Objasnite važnost održavanja pH vrednosti unutar optimalnog opsega za unos hranljivih materijala (obično oko pH 6-7).
- Vodite učenike u kalibraciji njihovih pH senzora koristeći rastvore kalibracije pH vrednosti.

Evo primera Arduino koda za automatizovani hidroponski sistem za navodnjavanje sa praćenjem i kontrolom pH vrednosti. Ovaj kod je osnovni okvir koji možete prilagoditi i proširiti prema vašem konkretnom postavkom i potrebama.

```
#include <Adafruit_ADS1015.h>
#include <Wire.h>
#define PH_SENSOR_PIN 0 // Analog pin for pH sensor
#define PUMP_PIN 12    // Digital pin for water pump
// Create an Adafruit ADS1015 ADC object
Adafruit_ADS1015 ads;
float currentpH;
float targetpH = 6.5; // Adjust this value to your desired pH level
void setup() {
 Serial.begin(9600);
 ads.begin();
 pinMode(PUMP_PIN, OUTPUT);
 digitalWrite(PUMP_PIN, LOW); // Initialize pump as off
}
void loop() {
 // Read pH value from pH sensor
 currentpH = readpH();
 // Display current pH value
 Serial.print("Current pH: ");
 Serial.println(currentpH);
 // Check and adjust pH level
 if (currentpH < targetpH) {
  // pH is too low, add pH up solution (adjust as needed)
  // Implement pH adjustment mechanism here
    // For example, you can control a peristaltic pump for adding pH up solution
```



```
digitalWrite(PUMP_PIN, HIGH); // Turn on the pump
 delay(1000); // Adjust the delay time as needed
 digitalWrite(PUMP_PIN, LOW); // Turn off the pump
 } else if (currentpH > targetpH) {
 // pH is too high, add pH down solution (adjust as needed)
 // Implement pH adjustment mechanism here
  // For example, you can control a peristaltic pump for adding pH down solution
 digitalWrite(PUMP_PIN, HIGH); // Turn on the pump
 delay(1000); // Adjust the delay time as needed
 digitalWrite(PUMP_PIN, LOW); // Turn off the pump
 }
 // Add code for irrigation control here (e.g., based on time intervals)
}
float readpH() {
 // Read pH value from the pH sensor and convert to pH scale
 int16_t rawValue = ads.readADC_SingleEnded(PH_SENSOR_PIN);
 float voltage = (rawValue * 0.1875) / 1000.0; // Convert to voltage
  float pHValue = 3.5 * voltage + 3.5; // Convert to pH scale (adjust calibration values as needed)
 return pHValue;
}
```

U ovom kodu:

Koristimo Adafruit ADS1015 biblioteku kako bismo interfejsirali sa ADS1015 ADC, koji se koristi za čitanje vrednosti pH senzora. Obezbedite da imate ovu biblioteku instaliranu u vašem Arduino IDE.
Funkcija setup() inicijalizuje serijsku komunikaciju za izlaz podataka, inicijalizuje ADC i konfiguriše pin za vodenu pumpu kao izlaz.
U funkciji loop() kontinuirano čitamo pH vrednost sa pH senzora koristeći funkciju readpH().
Upoređujemo trenutnu pH vrednost sa ciljnom pH vrednošću (targetpH) i prilagođavamo pH po potrebi. Trebalo bi da implementirate mehanizam za prilagođavanje pH vrednosti prema vašem specifičnom postavkom, što može uključivati kontrolu peristaltičke pumpe za dodavanje rastvora za podešavanje pH vrednosti (pH up ili pH down).
Dodatno, možete dodati kod za kontrolu vodene pumpe za navodnjavanje na osnovu vremenskih intervala ili drugih kriterijuma.
Napomena da ovaj kod pruža osnovni okvir i možda ćete morati da ga kalibrišete i podešavate prema vašem specifičnom senzoru i postavci pumpe, kao i prema željenim pH vrednostima i rasporedu navodnjavanja. Obezbedite sigurnosne mere kada radite sa hemikalijama i pumpama.

Dan 4

Postavljanje Eksperimenta i Praćenje (60 minuta):

- Dozvolite učenicima da postave svoje automatizovane hidroponske sisteme u stakleniku ili u učionici.
- Učenici treba da pokrenu sistem i prate rast biljaka, nivo hranljivih materijala i pH vrednosti.

Prezentacija Projekta i Diskusija (60 minuta):

- Svaka grupa predstavlja dizajn svog hidroponskog sistema, metodologiju i početne rezultate predavanju.
- Diskutujte o uticaju rastvora hranljivih materijala i kontrole pH vrednosti na rast biljaka.
- Podstaknite učenike da predlože optimizacije za svoje sisteme na osnovu početnih posmatranja.

Procena

Ocenite učenike na osnovu njihovog učešća, postavljanja sistema, kalibracije pH vrednosti, prikupljanja podataka, analize i kvaliteta njihovih prezentacija.

Domaći

Dodelite domaći zadatak učenicima koji podrazumeva istraživanje i prezentovanje stvarne primene hidroponike u poljoprivredi ili održivom uzgoju biljaka.

Zaključak

Zaključite lekciju sumiranjem ključnih hemijskih koncepata naučenih tokom časa i naglasite značaj hidroponike u održivoj poljoprivredi i proizvodnji hrane. Istaknite ulogu tehnologije (Arduino) u automatizaciji i optimizaciji hidroponskih sistema.

FROM GAME TO STEM

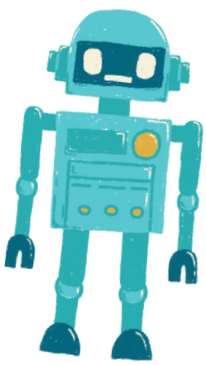
Co-funded by the Erasmus+ Programme of the European Union

Lekcija 6
Nastava hemije sa Arduinom

Izvođenje hemijskih reakcija sa Arduinom

Trajanje:

3 perioda



Cilj:

- Učenici će razumeti principe hemijskih reakcija i njihovu relevanciju u svakodnevnom životu.
- Učenici će dizajnirati i sprovoditi eksperimente kako bi posmatrali i merili hemijske reakcije.
- Učenici će programirati sistem za praćenje temperature baziran na Arduino platformi kako bi prikupili i analizirali podatke o hemijskim reakcijama.

Materijal:

- Arduino ploče (jedna po učeniku ili grupi)
- Senzori temperature (npr. DS18B20 ili LM35)
- Hemijske supstance za eksperimente (npr. soda bikarbona, sirće)
- Posude za reakcije (npr. bekeri, epruvete)
- Sklopni kablovi
- USB kablovi za programiranje
- Računari sa instaliranim Arduino IDE
- Projektor ili tabla za demonstracije
- Zaštitne naočare i laboratorijske mantile

Aktivnosti

Dan 1

<u>Uvod u hemijske reakcije (15 minuta):</u>

- Započnite lekciju uvodnom prezentacijom pojma hemijskih reakcija i njihove uloge u svakodnevnom životu.
- Diskutujte o značaju hemijskih reakcija u različitim oblastima, uključujući hemiju, biologiju i industriju.

<u>Osnove hemijskih reakcija (30 minuta):</u>

- Dublje se upustite u osnove hemijskih reakcija, uključujući reaktante, produkte i princip očuvanja mase.
- Pružite primere uobičajenih hemijskih reakcija i njihovih primena.
- Naglasite protokole bezbednosti prilikom rukovanja hemikalijama u eksperimentima.

<u>Uvod u Arduino i senzore (20 minuta):</u>

- Predstavite Arduino kao platformu za prikupljanje podataka i objasnite njegove komponente (mikrokontroler, senzori).
- Pokažite primere temperaturnih senzora i kako ih povezati sa Arduino platformom za prikupljanje podataka.

<u>Priprema za praktičnu aktivnost (45 minuta):</u>

- Dodelite učenicima Arduino ploče, temperaturne senzore, hemikalije i posude za reakcije.
- Instruirajte učenike da razmišljaju i planiraju svoje eksperimente hemijskih reakcija, sa fokusom na promene temperature.
- Razgovarajte o merama bezbednosti i pravilima laboratorije.

Dan 2

Postavljanje eksperimenta i prikupljanje podataka (60 minuta):

- Počnite drugi dan tako što ćete dozvoliti učenicima da postave svoje eksperimente hemijskih reakcija.
- Učenici treba da mešaju odabrane hemikalije u posudama za reakcije i postave temperaturne senzore.
- Instruirajte učenike da napišu Arduino kod za praćenje temperature i prikupljanje podataka.

Analiza podataka i diskusija (30 minuta):

- Vodite učenike u analizi svojih podataka, tražeći promene temperature tokom hemijskih reakcija.
- Diskutujte o značaju promena temperature u hemijskim reakcijama i konceptima eksotermnih i endotermnih reakcija.
- Ohrabrite učenike da izvuku zaključke na osnovu svojih nalaza.

Dan 3

Programiranje sistema za praćenje (60 minuta):

- Naredite učenicima da dorade svoj Arduino kod za prikupljanje i analizu podataka.
- Učenici treba da programiraju Arduino da beleži podatke o temperaturi u određenim vremenskim intervalima i da ih prikazuje grafički (npr. na LCD ekranu ili na Serial Monitoru).

Prezentacija projekta i diskusija (60 minuta):

- Svaka grupa treba da prezentuje svoj eksperiment sa hemijskom reakcijom, metodologiju i rezultate pred razredom.
- Ohrabrite učenike da objasne svoja zapažanja i posledice promena temperature u hemijskim reakcijama.
- Facilitirajte diskusiju u razredu o primenama hemijskih reakcija u stvarnim situacijama u različitim oblastima.

Evo primera Arduino koda koji možete koristiti za praćenje temperature tokom eksperimenta hemijske reakcije. Ovaj kod čita podatke o temperaturi sa DS18B20 temperaturnog senzora i prikazuje ih na Serial Monitoru. Možete prilagoditi i proširiti ovaj kod prema vašem specifičnom eksperimentu.

```
#include <OneWire.h>
#include <DallasTemperature.h>
// Data wire is plugged into digital pin 2
#define ONE_WIRE_BUS 2
// Create a OneWire object
OneWire oneWire(ONE_WIRE_BUS);
// Pass the OneWire reference to Dallas Temperature sensor
DallasTemperature sensors(&oneWire);
```

```
void setup() {
 // Initialize serial communication for data output
 Serial.begin(9600);

 // Start the temperature sensor library
 sensors.begin();
}

void loop() {
 // Request temperature readings
 sensors.requestTemperatures();

 // Read temperature in Celsius
 float temperatureC = sensors.getTempCByIndex(0);

 // Display temperature on the Serial Monitor
 Serial.print("Temperature: ");
 Serial.print(temperatureC);
 Serial.println(" °C");

 // Add code here for data storage or further analysis

 // Delay before the next temperature reading (adjust as needed)
 delay(1000); // 1-second delay
}
```

U ovom kodu:

Koristimo Dallas Temperature biblioteku za interakciju sa DS18B20 temperaturnim senzorom. Obezbedite da imate ovu biblioteku instaliranu u vašem Arduino IDE.
Funkcija setup() inicijalizuje serijsku komunikaciju za izlaz podataka i pokreće biblioteku za temperaturni senzor.
U funkciji loop() program neprestano zahteva i čita podatke o temperaturi sa senzora koristeći sensors.getTempCByIndex(0). Broj 0 označava prvi (a u ovom slučaju, jedini) temperaturni senzor povezan.

Podaci o temperaturi se prikazuju na Serial Monitoru sa pauzom od 1 sekunde između očitavanja. Možete prilagoditi vreme pauze da biste kontrolisali frekvenciju prikupljanja podataka.

Možete izmeniti ovaj kod da uključite dodatne senzore za različite eksperimente, implementirate skladištenje podataka na SD karticu ili kreirate grafičke prikaze podataka na LCD ekranu, u zavisnosti od vaših specifičnih zahteva.

Procena

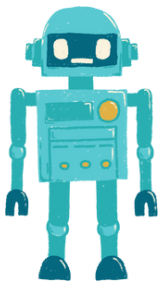
Ocjenjujte učenike na osnovu njihovog učešća u eksperimentu, prikupljanja podataka, analize i kvaliteta njihovih prezentacija.

Domaći

Dodelite domaći zadatak koji zahteva od učenika da istraže i prezentuju stvarnu primenu hemijskih reakcija u određenoj industriji ili oblasti nauke.

Zaključak

Zaključite lekciju sažimanjem ključnih hemijskih koncepata koje su učenici naučili i naglašavanjem važnosti hemijskih reakcija u razumevanju prirodnih procesa i tehnoloških dostignuća. Istaknite ulogu tehnologije (Arduino) u naučnim istraživanjima.



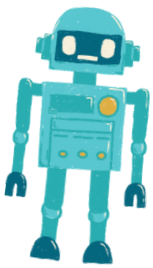
FROM GAME TO STEM

Co-funded by the
Erasmus+ Programme
of the European Union

Lekcija 7
Nastava biologije sa Arduinom

Ispitivanje termičke provodljivosti pomoću Arduino termometara

Trajanje:
3 perioda



Cilj:

- Učenici će razumeti pojam termičke provodljivosti i njegovu važnost.
- Učenici će osmisliti i sprovesti eksperiment kako bi odredili termičku provodljivost različitih materijala.
- Učenici će koristiti Arduino termometre za prikupljanje podataka o temperaturi i analizirati svoje rezultate."

Materijal:

- Arduino ploče (po jedna po učeniku ili grupi)
- Senzori temperature (npr. DS18B20 ili LM35)
- Breadboard-ovi
- Jumper žice
- Razni materijali za ispitivanje provodljivosti (npr. metali, plastika, drvo, staklo)
- Izolacioni materijali (npr. pena, tkanina)
- Topla voda ili izvor toplote
- Hladna voda ili led
- Stop sat ili tajmer
- USB kablovi za programiranje
- Računari sa instaliranim Arduino IDE
- Projektor ili tabla za demonstracije
- Zaštitne naočare i rukavice (za rukovanje vrućim materijalima)"

Aktivnosti

Dan 1

Uvod u Termičku Provodljivost (15 minuta):

- Započnite čas uvodom u pojam termičke provodljivosti i zašto je ona važna u svakodnevnom životu.
- Razgovarajte o primenama termičke provodljivosti u stvarnom svetu, kao što su kuvanje, građevinski materijali i inženjering.

Prenos toplote i Izolacija (30 minuta):

- Objasnite različite metode prenosa toplote (provodljivost, konvekcija, zračenje) i fokusirajte se na provodljivost, koja je relevantna za eksperiment.
- Razgovarajte o izolacionim materijalima i njihovoj ulozi u smanjenju prenosa toplote.
- Naglasite potrebu za kontrolisanim eksperimentom kako biste izmerili termičku provodljivost.

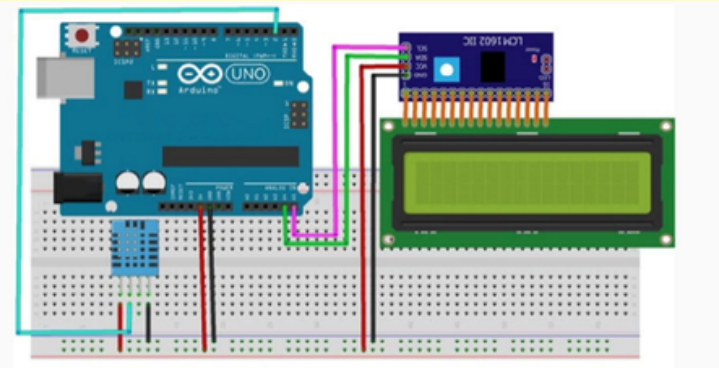
Uvod u Arduino Termometre (20 minuta):

- Predstavite Arduino kao platformu za prikupljanje podataka i objasnite njegove komponente (mikrokontroler, senzori).
- Objasnite upotrebu senzora temperature i kako se mogu povezati sa Arduino pločama.
- Pokažite primere prikupljanja podataka o temperaturi i vizualizacije sa Arduino.

Priprema za Aktivnost sa Praktičnim Radom (45 minuta):

- Dajte učenicima Arduino ploče, senzore temperature, breadboard-ove i jumper žice.
- Objasnite eksperimentalni postav: Svaka grupa će testirati različite materijale kako bi odredila njihovu termičku provodljivost.
- Nastavite učenike da razmišljaju i planiraju svoje eksperimente, uključujući kako da kontrolišu promenljive i prikupe podatke.

Šema:



Дан 2

Priprema za Eksperiment i Prikupljanje Podataka (60 minuta):

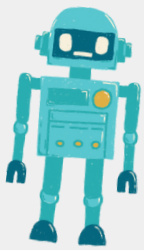
- Drugog dana, omogućite učenicima da postave svoje eksperimente.
- Učenici trebaju pričvrstiti senzor temperature na odabrane materijale i pripremiti posude sa toplom i hladnom vodom.
- Nastavite učenike da počnu prikupljati podatke o temperaturi koristeći Arduino i beleže početne temperature.
- Neka učenici mere i zabeleže vreme potrebno za promenu temperature svakog materijala.

Analiza Podataka i Diskusija (30 minuta):

- Vodite učenike u analizi svojih podataka, računanju brzine promene temperature i upoređivanju provodljivosti različitih materijala.
- Razgovarajte o pojmu termičkog otpora i kako se odnosi na provodljivost.
- Ohrabrite učenike da identifikuju koji materijal je najbolji provodnik toplote, a koji najgori na osnovu svojih nalaza.

Evo primera Arduino koda koji možete koristiti za eksperiment kako biste merili i upoređivali termičku provodljivost različitih materijala koristeći senzor temperature (npr. DS18B20). Ovaj kod će prikupljati podatke o temperaturi sa senzora i omogućiti vam da izračunate brzinu promene temperature za svaki materijal.

```
#include <OneWire.h>
#include <DallasTemperature.h>
// Data wire is plugged into digital pin 2
#define ONE_WIRE_BUS 2
// Create a OneWire object
OneWire oneWire(ONE_WIRE_BUS);
// Pass the OneWire reference to Dallas Temperature sensor
DallasTemperature sensors(&oneWire);
void setup() {
 // Initialize serial communication for data output
 Serial.begin(9600);
 // Start the temperature sensor library
 sensors.begin();
}
void loop() {
```



```
// Initialize variables for temperature measurements
 float initialTemp, finalTemp;
 unsigned long startTime, endTime;
 float rateOfChange;
 // Wait for the user to start the experiment (e.g., press a button)
while (!digitalRead(3)) {
  // Wait for button press
 }
 delay(1000); // Debounce delay
 // Measure initial temperature
 sensors.requestTemperatures(); // Request temperature readings
 initialTemp = sensors.getTempCByIndex(0); // Get temperature in Celsius
 // Record the start time
 startTime = millis();
 // Wait for the temperature to stabilize (e.g., 5 seconds)
 delay(5000);
 // Measure final temperature
 sensors.requestTemperatures();
 finalTemp = sensors.getTempCByIndex(0);
 // Record the end time
 endTime = millis();
 // Calculate the rate of temperature change (°C per second)
  rateOfChange = (finalTemp - initialTemp) / ((endTime - startTime) /
1000.0);
 // Output results to the Serial Monitor
 Serial.print("Initial Temperature: ");
 Serial.print(initialTemp);
 Serial.println(" °C");
 Serial.print("Final Temperature: ");
 Serial.print(finalTemp);
 Serial.println(" °C");
 Serial.print("Rate of Change: ");
 Serial.print(rateOfChange);
 Serial.println(" °C/s");
 // Wait for user input (e.g., press a button) to proceed to the next material
 Serial.println("Press a button to test the next material.");
 while (!digitalRead(3)) {
  // Wait for button press
 }
 delay(1000); // Debounce delay
}
```

U ovom kodu:

Koristimo Dallas Temperature biblioteku za komunikaciju sa DS18B20 senzorom temperature. Proverite da imate ovu biblioteku instaliranu u vašem Arduino IDE.
Definišemo digitalni pin (npr. pin 2) na koji je povezan data wire DS18B20 senzora.
U setup() funkciji, inicijalizujemo serijsku komunikaciju za ispisivanje podataka i pokrećemo biblioteku za temperaturni senzor.
U loop() funkciji, program čeka pritisak dugmeta za početak eksperimenta. Temperatura se meri pre i posle određenog vremenskog perioda (npr. 5 sekundi) kako biste izračunali brzinu promene temperature.
Brzina promene se računa kao razlika u temperaturi podeljena sa vremenom koje je potrebno da se ta promena desi. Ovo vam daje brzinu u °C po sekundi.
Rezultati se ispisuju na Serial Monitor-u, uključujući početnu temperaturu, krajnju temperaturu i brzinu promene.
Nakon završetka eksperimenta za jedan materijal, možete pritisnuti dugme kako biste prešli na sledeći materijal.
Zapamtite da pravilno povežete DS18B20 senzor sa Arduino pločom i obezbedite da je dugme povezano sa digitalnim pinom (npr. pin 3) za pokretanje eksperimenta. Prilagodite kod prema potrebi na osnovu vašeg specifičnog postava.

Дан 3

Projektna Prezentacija i Diskusija (60 minuta):

- Svaka grupa predstavlja svoj eksperimentalni postav, metodologiju i rezultate pred razredom.
- Ohrabrite učenike da objasne svoja zapažanja, diskutuju o mogućim izvorima grešaka i predlože poboljšanja.
- Osmislite diskusiju u razredu o stvarnim implikacijama njihovih nalaza, kao što su izbor materijala za termalnu izolaciju ili materijala za toplinske izmenjivače.

Procena

Ocenjujte učenike na osnovu njihovog učešća u eksperimentu, prikupljanja podataka, analize i kvaliteta njihovih prezentacija.

Domaći

Zadajte domaći zadatak učenicima koji će ih podstaći da istraže i prezentuju stvarnu primenu termičke provodljivosti u inženjeringu ili naukama o materijalima.

Zaključak

Zaključite čas sumirajući ključne koncepte koje su učenici naučili i naglašavajući važnost razumevanja termičke provodljivosti u svakodnevnom životu i tehnološkim primenama.

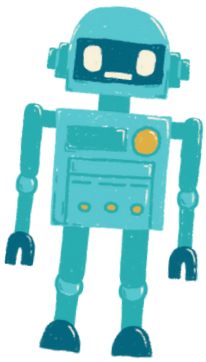
FROM GAME TO STEM

Co-funded by the Erasmus+ Programme of the European Union

Lekcija 8
Nastava biologije uz Arduino

Istraživanje fotosinteze i rasta biljaka sa Arduinom

Trajanje:
3 perioda



Cilj:

- Učenici će razumeti proces fotosinteze i njegov značaj za rast biljaka.
- Učenici će osmisliti i sprovesti eksperiment kako bi istražili uticaj faktora sredine na fotosintezu.
- Učenici će programirati sistem zasnovan na Arduino platformi za praćenje i prikupljanje podataka o rastu biljaka.

Materijal:

- Arduino ploče (jedna po učeniku ili grupi)
- Senzori (npr. senzor svetla, senzor temperature, senzor vlage)
- LED svetla za uzgoj (opciono)
- Saksije sa biljkama ili semenkama
- Zemlja za sadnju i posude za saksije
- Jumper žice
- USB kablovi za programiranje
- Računari sa instaliranim Arduino IDE
- Projektor ili tabla za demonstracije
- Zemlja za sadnju, voda i drugi materijali za negu biljaka.

Aktivnosti

Dan 1

<u>Uvod u Fotosintezu (15 minuta):</u>

- Započnite čas uvodom u pojam fotosinteze i njen značaj za rast biljaka.
- Razgovarajte o hemijskoj jednačini fotosinteze i ulozi svetla, ugljen-dioksida i vode u procesu.

<u>Uticanje Faktora Sredine na Rast Biljaka (30 minuta):</u>

- Objasnite kako različiti faktori sredine, kao što su intenzitet svetla, temperatura i vlaga, mogu uticati na fotosintezu i rast biljaka.
- Diskutujte zašto su ovi faktori ključni za zdrav razvoj biljaka.
- Naglasite potrebu za kontrolisanim eksperimentima kako biste proučavali ove faktore.

<u>Uvod u Arduino i Senzore (20 minuta):</u>

- Predstavite Arduino kao platformu za prikupljanje podataka i objasnite njegove komponente (mikrokontroler, senzori).
- Pokažite primere senzora koji se često koriste u praćenju životne sredine i nege biljaka.
- Objasnite ulogu senzora u prikupljanju podataka za eksperimente.

<u>Priprema za Aktivnost sa Praktičnim Radom (45 minuta):</u>

- Dajte učenicima Arduino ploče, senzore (npr. senzor svetla, senzor temperature, senzor vlage) i LED svetla za uzgoj (opciono).
- Nastavite učenike da razmišljaju i planiraju svoje eksperimente sa rastom biljaka, fokusirajući se na jedan faktor sredine.
- Neka učenici pripreme saksije sa zemljom i posade seme ili male biljke u njih.

Dan 2

Experiment Setup and Data Collection (60 minutes):

- Započnite drugi dan tako što ćete omogućiti učenicima da postave svoje eksperimente.
- Učenici treba da postave senzore u okolinu biljaka, povežu ih sa Arduino pločom i postave LED svetla za uzgoj (ako se koriste).
- Nastavite učenike da prikupljaju podatke vezane za izabrani faktor sredine, kao što su intenzitet svetla ili temperatura.
- Naglasite važnost tačnog i redovnog beleženja podataka.

Analiza Podataka i Diskusija (30 minuta):

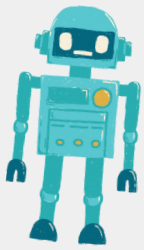
- Vodite učenike u analizi svojih podataka, tražeći trendove ili obrasce koji se odnose na rast biljaka i izabrani faktor.
- Diskutujte o uticaju faktora na fotosintezu i razvoj biljaka.
- Ohrabrite učenike da izvuku zaključke iz svojih nalaza.

Dan 3

Programming the Monitoring System (60 minutes):

- Nastavite učenike da napišu Arduino kod za praćenje i prikupljanje podataka sa senzora.
- Učenici treba da programiraju Arduino da beleži podatke u određenim intervalima (npr. svaki sat).
- Razgovarajte o tome kako koristiti biblioteke za dobijanje podataka sa senzora i kako skladištiti podatke.

Evo primera Arduino koda za jednostavan sistem za praćenje životne sredine koji meri i beleži podatke o intenzitetu svetla pomoću senzora svetla.



```
#include <Wire.h>
#include <Adafruit_Sensor.h>
#include <Adafruit_TSL2591.h>
Adafruit_TSL2591 tsl = Adafruit_TSL2591(2591);
void setup() {
 Serial.begin(9600);
  // Initialize the light sensor
 if(!tsl.begin()) {
   Serial.println("Light sensor not found. Check wiring.");
   while(1);
 }
 tsl.setGain(TSL2591_GAIN_LOW);    // Adjust the gain (options: LOW, MED, HIGH, MAX)
 tsl.setTiming(TSL2591_INTEGRATIONTIME_100MS); // Adjust the integration time (options:
100, 200, 300, 400, 500, 600)
}
void loop() {
 // Read and print light intensity data
 uint16_t luminance = tsl.getLuminosity(TSL2591_VISIBLE);
 Serial.print("Light Intensity (Lux): ");
 Serial.println(luminance);
    // Add code here to log data to an SD card, display on an LCD, or transmit to a
computer/server.
 // Wait for a specific time interval (e.g., 1 hour)
 delay(3600000); // Adjust the delay time as needed
}
```

U ovom kodu:

Koristimo Adafruit TSL2591 biblioteku za komunikaciju sa TSL2591 senzorom svetla. Proverite da imate ovu biblioteku instaliranu u vašem Arduino IDE.
U setup() funkciji inicijalizujemo serijsku komunikaciju za ispisivanje podataka i postavljamo senzor svetla. Takođe konfigurišemo pojačanje (gain) i vreme integracije senzora prema vašim zahtevima.
U loop() funkciji, program neprekidno čita podatke o intenzitetu svetla (u luksima) sa senzora koristeći tsl.getLuminosity(TSL2591_VISIBLE).
Možete dodati kod unutar loop-a da biste beležili podatke na SD karticu, prikazivali ih na LCD ekranu ili ih slali računaru ili serveru za dalju analizu. Na primer, možete koristiti modul za SD karticu da biste skladištili podatke lokalno.
Zadržavanje (delay) na kraju loop-a postavljeno je da čeka određeni vremenski interval (npr. 1 sat) pre nego što uzme sledeće očitavanje. Možete prilagoditi vreme čekanja kako biste kontrolisali frekvenciju prikupljanja podataka.

Ovaj kod pruža osnovni okvir za praćenje intenziteta svetla, a možete ga proširiti dodavanjem više senzora za praćenje dodatnih faktora životne sredine kao što su temperatura i vlažnost. Ne zaboravite prilagoditi kod kako bi odgovarao konkretnim senzorima i hardveru koje koristite u svom eksperimentu.

Prezentacija Projekta i Diskusija (60 minuta):

- Svaka grupa će predstaviti svoj eksperiment sa rastom biljaka, postavku, metodologiju i rezultate pred razredom.
- Ohrabrite učenike da objasne svoja zapažanja i implikacije njihovih nalaza za negu biljaka i poljoprivredu.
- Organizujte diskusiju u razredu o značaju fotosinteze u ekosistemu i o tome kako tehnologija (Arduino) može pomoći u naučnim istraživanjima.

Procena

Ocenjujte učenike na osnovu njihovog učešća u eksperimentu, prikupljanja podataka, analize i kvaliteta njihovih prezentacija.

Domaći

Dodelite domaći zadatak učenicima da istraže i prezentuju stvarnu primenu fotosinteze i praćenja životne sredine u poljoprivredi ili ekologiji.

Zaključak

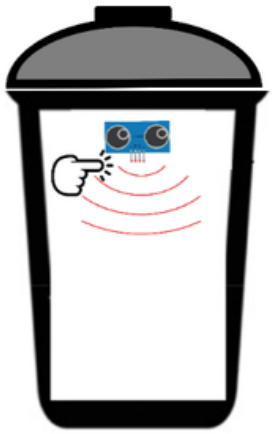
Zaključite čas sumirajući ključne biološke koncepte koje su učenici naučili i naglašavajući važnost fotosinteze i faktora sredine u rastu biljaka. Istaknite ulogu tehnologije u unapređenju naučnog razumevanja.

FROM GAME TO STEM

Co-funded by the
Erasmus+ Programme
of the European Union

Lekcija 9
Ekologija sa Arduinom

Izgradnja pametne kante sa Arduino



Trajanje:

3 perioda

Cilj:

- Učenici će razumeti osnove programiranja sa Arduino platformom i njene primene u automatizaciji.
- Učenici će naučiti o ultrazvučnim senzorima i kako se koriste za detekciju objekata.
- Učenici će izgraditi prototip pametne kante i programirati je da otvara poklopac kada se detektuje objekat.
- Učenici će istražiti ekološke koristi sistema za pametno upravljanje otpadom.

Materijal:

- Arduino Uno ploče (jedna po učeniku ili grupi)
- Ultrazvučni senzor HC-SR04 (jedan po učeniku ili grupi)
- Servomotori (jedan po učeniku ili grupi)
- Breadboard-ovi i sklopke sa žicama
- Mala kartonska kutija ili kontejner (za kantu)
- Kartonski ili plastični poklopac (za simulaciju poklopca kante)
- USB kablovi za programiranje
- Računari sa instaliranim Arduino IDE
- Projektor ili tabla za demonstracije

Aktivnosti

Dan 1

Uvod u Arduino i Elektroniku (15 minuta):

- Započnite čas uvodom u Arduino kao mikrokontrolersku platformu koja se koristi za razne projekte.
- Diskutujte značaj elektronike i automatizacije u savremenoj tehnologiji.

Ultrazvučni Senzori i Detekcija Objekata (30 minuta):

- Objasnite princip rada ultrazvučnih senzora i kako se koriste za merenje udaljenosti.
- Razgovarajte o komponentama HC-SR04 ultrazvučnog senzora (predajnik i prijemnik).
- Ilustrujte kako se ultrazvučni senzori mogu koristiti za detekciju objekata i merenje udaljenosti.

Uvod u Servomotore (20 minuta):

- Predstavite servomotore i njihove primene u kontroli mehaničkih pokreta.
- Pokažite primere kako se servomotori mogu koristiti u projektima kao što je otvaranje poklopca.

Priprema za Aktivnost sa Praktičnim Radom (45 minuta):

- Dajte svakoj grupi Arduino Uno, ultrazvučni senzor, servomotor, breadboard i jumper žice.
- Nastavite učenike da sastave prototip pametne kante, postavljajući ultrazvučni senzor i servomotor na odgovarajući način.

Dan 2

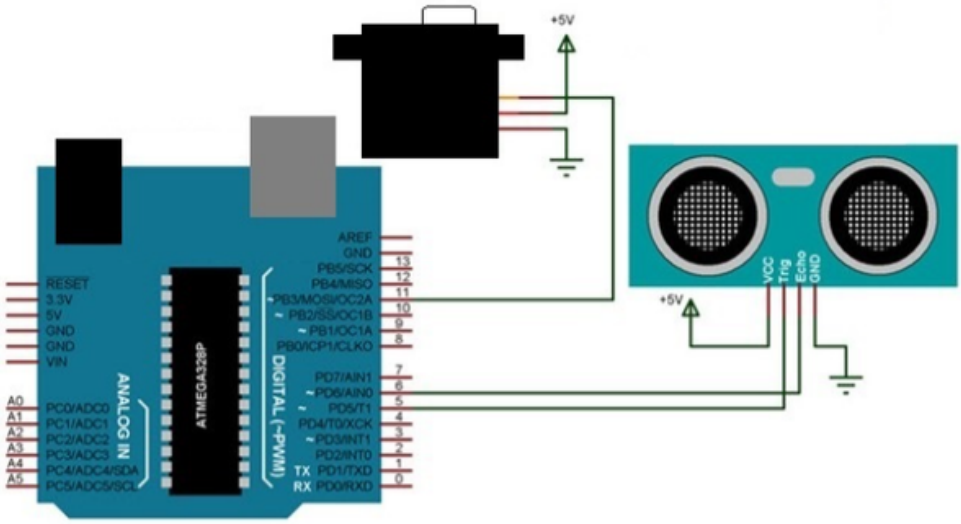
Osnove Arduino Programiranja (30 minuta):

- Naučite učenike osnovama Arduino programiranja, uključujući setup(), loop() i pinMode().
- Priložite primere osnovnog Arduino koda za treptanje LED diode.

Programiranje Pametne Kante (60 minuta):

- Vodite učenike u pisanju Arduino koda za kontrolu Pametne Kante na osnovu očitavanja ultrazvučnog senzora.
- Objasnite logiku za otvaranje poklopca kada se detektuje objekat unutar određenog opsega.

Šema:



Evo primera Arduino koda za Pametnu Kantu sa ultrazvučnim senzorom (HC-SR04) i servomotorom. Ovaj kod omogućava servomotoru da otvori poklopac kante kada se detektuje objekat unutar određenog opsega.

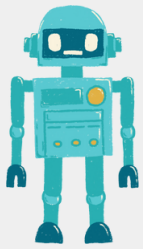
```
#include <Servo.h>

#define TRIGGER_PIN 9
#define ECHO_PIN 10
#define SERVO_PIN 11

Servo myservo; // Create a Servo object
int distance; // Variable to store distance measured by the Ultrasonic sensor

void setup() {
 myservo.attach(SERVO_PIN); // Attach the Servo to the specified pin
 pinMode(TRIGGER_PIN, OUTPUT);
 pinMode(ECHO_PIN, INPUT);
 Serial.begin(9600); // Initialize serial communication for debugging
}
```

```
void loop() {
 // Send a brief pulse to trigger the Ultrasonic sensor
 digitalWrite(TRIGGER_PIN, LOW);
 delayMicroseconds(2);
 digitalWrite(TRIGGER_PIN, HIGH);
 delayMicroseconds(10);
 digitalWrite(TRIGGER_PIN, LOW);
 // Read the duration of the echo pulse and calculate the distance
 duration = pulseIn(ECHO_PIN, HIGH);
 distance = (duration / 2) / 29.1; // Calculate distance in centimeters
 // Check if an object is within the specified range (adjust as needed)
 if (distance < 20) { // You can adjust the distance threshold here
  // If an object is detected, open the flap
   myservo.write(90); // Rotate the Servo to open the flap (adjust the angle as needed)
  delay(1000); // Wait for 1 second
  myservo.write(0); // Rotate the Servo back to close the flap
 }
 // Print the distance to the Serial Monitor for debugging
 Serial.print("Distance: ");
 Serial.print(distance);
 Serial.println(" cm");
 // Add a delay between readings to prevent rapid triggering
 delay(1000); // You can adjust the delay time as needed
}
```



U ovom kodu:

Uključujemo Servo biblioteku i definišemo brojeve pinova za okidač i ekho ultrazvučnog senzora, kao i za kontrolni pin servomotora.
U setup() funkciji, povezujemo servomotor sa određenim pinom i postavljamo okidački pin kao izlaz i ekho pin kao ulaz. Takođe inicijalizujemo serijsku komunikaciju za potrebe debagovanja.
loop() funkcija ponavlja sledeće korake:
Šalje kratki impuls ultrazvučnom senzoru da bi pokrenuo merenje udaljenosti.
Meri trajanje ekho impulsa i izračunava udaljenost u centimetrima.
Proverava da li je objekat unutar određenog opsega (20 cm u ovom primeru) i, ako jeste, otvara poklopac kante tako što rotira servomotor na određeni ugao (90 stepeni).
Ispisuje udaljenost na serijski monitor u svrhu debagovanja.
Dodaje pauzu između očitavanja kako bi sprečio brzo okidanje.
Možete prilagoditi prag udaljenosti, ugao servomotora i vreme pauze kako biste se prilagodili vašem specifičnom postavci i zahtevima.

Testing and Troubleshooting (30 minutes):

- Nastavite učenike da testiraju svoje prototipe pametnih kanti i da rešavaju eventualne probleme sa kodom ili hardverom.
- Ohrabrite eksperimentisanje sa različitim udaljenostima detekcije i uglovima servomotora.

Dan 3

Project Presentation and Discussion (60 minutes):

- Svaka grupa će predstaviti svoj projekat pametne kante pred razredom, objašnjavajući dizajn, komponente i kod.
- Diskutujte o ekološkim prednostima sistema za pametno upravljanje otpadom i njihovom potencijalnom uticaju na smanjenje otpada.

Otvorena Diskusija i Buduća Unapređenja (30 minuta):

- Vodite razrednu diskusiju o potencijalnim unapređenjima Pametne Kante i drugim aplikacijama slične tehnologije.
- Ohrabrite učenike da razmišljaju o idejama za dalje unapređenje rešenja za upravljanje otpadom.

Procena

Ocenjujte učenike na osnovu njihovog učešća, funkcionalnosti projekta, razumevanja Arduino programiranja i sposobnosti da reše probleme.

Domaći

Zadajte domaći zadatak učenicima da istraže i prezentuju stvarne primere sistema za pametno upravljanje otpadom i njihov uticaj na održivost.

Zaključak

Zaključite čas sumirajući ključne koncepte koje su učenici naučili, ističući presecanje tehnologije i ekoloških rešenja, i ohrabrite učenike da kritički razmišljaju o primeni tehnologije za pozitivne promene.

FROM GAME TO STEM

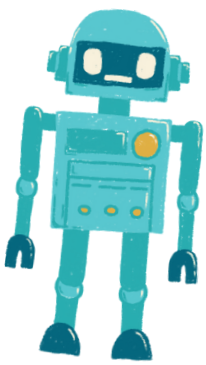
Co-funded by the Erasmus+ Programme of the European Union

Lekcija 10
Ekologija sa Arduinom

Istraživanje Kvaliteta Vazduha sa Arduinom

Trajanje:

1 period



Cilj:

- Učenici će saznati o značaju kvaliteta vazduha za ekološko zdravlje.
- Učenici će razumeti kako zagađivači vazduha mogu uticati na ekosisteme i ljudsko zdravlje.
- Učenici će napraviti jednostavan senzor kvaliteta vazduha zasnovan na Arduino platformi i prikupljati podatke.
- Učenici će diskutovati o značaju praćenja kvaliteta vazduha za ekološko blagostanje.

Materijal

- Arduino Uno ploče (jedna po učeniku ili grupi)
- Modul senzora kvaliteta vazduha (npr. MQ-135)
- Jumper žice
- USB kablovi za programiranje
- Računari sa instaliranim Arduino IDE
- Projektor ili tabla za demonstracije

Aktivnosti

Dan 1

Uvod u Kvalitet Vazduha (10 minuta):

- Započnite čas diskutujući o važnosti čistog vazduha i njegovog uticaja na ekosisteme.
- Objasnite kako zagađivači vazduha mogu uticati kako na prirodne okoline, tako i na ljudsko zdravlje.

Senzori Kvaliteta Vazduha (20 minuta):

- Uvedite senzore kvaliteta vazduha i njihovu ulogu u praćenju zagađenja vazduha.
- Objasnite osnovno delovanje senzora kvaliteta vazduha i kako mere različite gasove.

Osnove Arduino (10 minuta):

- Predstavite Arduino kao mikrokontrolersku platformu za prikupljanje podataka.
- Pokažite učenicima komponente Arduino ploče i osnove pisanja i učitavanja koda.

Priprema za Praktični Rad (15 minuta):

- Dajte svakoj grupi Arduino Uno, modul senzora kvaliteta vazduha i jumper žice.
- Nastavite učenike da sastave senzor kvaliteta vazduha i povežu ga sa Arduino.

Izgradnja i Programiranje Senzora Kvaliteta Vazduha (20 minuta):

- Vodite učenike u izgradnji njihovog sistema za merenje kvaliteta vazduha zasnovanog na Arduino platformi.
- Pokažite učenicima kako da napišu Arduino kod za prikupljanje podataka o kvalitetu vazduha sa senzora.

Evo jednostavnog primera Arduino koda za praćenje kvaliteta vazduha pomoću MQ-135 senzora kvaliteta vazduha. Ovaj kod čita podatke sa senzora i prikazuje ih na Serial Monitor-u. Obezbedite da imate odgovarajuće biblioteke instalirane u svom Arduino IDE-u, jer senzor MQ-135 može zahtevati kalibraciju za precizne rezultate.

```
// Include necessary libraries
#include <Wire.h>
#include <Adafruit_Sensor.h>
#include <Adafruit_MQ135.h>

// Define the analog pin where the MQ-135 sensor is connected
#define MQ135_PIN A0

// Create an instance of the Adafruit MQ135 class
Adafruit_MQ135 mq135(MQ135_PIN);

void setup() {
 // Initialize serial communication for data output
 Serial.begin(9600);
}

void loop() {
 // Read the MQ-135 sensor's data
 float airQuality = mq135.readCO2();

 // Print the air quality data to the Serial Monitor
 Serial.print("Air Quality: ");
 Serial.print(airQuality);
 Serial.println(" ppm");

 // Add a delay before the next reading
 delay(2000); // Adjust the delay time as needed
}
```

U ovom kodu:

Uključujemo potrebne biblioteke, uključujući Adafruit Sensor i Adafruit MQ135 biblioteke, koje se često koriste za rad sa MQ-135 senzorom kvaliteta vazduha. Obezbedite da imate ove biblioteke instalirane u svom Arduino IDE-u.
Definišemo analogni pin (A0 u ovom primeru) na koji je MQ-135 senzor povezan na Arduino.
Kreiramo instancu klase Adafruit_MQ135 za interakciju sa senzorom.
U setup() funkciji inicijalizujemo serijsku komunikaciju za ispis podataka na Serial Monitor.
U loop() funkciji neprekidno čitamo podatke o kvalitetu vazduha sa MQ-135 senzora koristeći mq135.readCO2(). Podaci predstavljaju koncentraciju CO2 u delovima po milionu (ppm).
Podaci o kvalitetu vazduha se ispisuju na Serial Monitor, a postoji pauza od 2 sekunde (koja se može prilagoditi) pre nego što se uzme sledeće očitavanje.
Napomena da MQ-135 senzor može zahtevati kalibraciju za precizna očitavanja, i kod koji je ovde pružen služi kao osnovni primer. Zavisno o vašoj specifičnoj aplikaciji i potrebama za kalibracijom, možda ćete morati prilagoditi kod i proces kalibracije.

Sakupljanje podataka i diskusija (20 minuta):

- Nastavnicima naložite da prikupe podatke o kvalitetu vazduha pokretanjem svojih senzora u različitim okruženjima (npr. unutra, blizu puta, u vrtu).
- Nastavite učenike da zabeleže i podele svoje podatke sa časom.
- Vodite razrednu diskusiju o razlikama u podacima o kvalitetu vazduha i potencijalnim ekološkim posledicama.

Procena

Ocenjujte učenike na osnovu njihovog učešća, prikupljanja podataka i sposobnosti da diskutuju o uticaju kvaliteta vazduha na ekološke sisteme.

Domaći

Zadajte domaći zadatak učenicima da istraže i prezentuju stvarne primere ekoloških problema povezanih sa zagađenjem vazduha i značaj praćenja kvaliteta vazduha u ublažavanju ovih problema.

Zaključak

Zadajte domaći zadatak učenicima da istraže i prezentuju stvarne primere ekoloških problema povezanih sa zagađenjem vazduha i značaj praćenja kvaliteta vazduha u ublažavanju ovih problema.

FROM GAME TO STEM

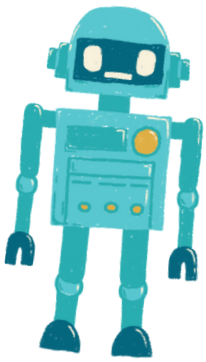
With the support of the Erasmus+ Programme of the European Union

Lekcja 1
Lekcja matematyki Arduino

Trygonometria z Arduino

Czas trwania:

3 okresy



Cel:

- Studenci będą rozumieć podstawowe pojęcia trygonometryczne, w tym sinus, cosinus i tangens.
- Studenci będą stosować funkcje trygonometryczne do rozwiązywania problemów świata rzeczywistego.
- Uczestnicy kursu zaprogramują Arduino w celu stworzenia prostego urządzenia do pomiaru kąta wykorzystującego serwomotor.

Materiały:

- Tablice Arduino (po jednej na ucznia lub grupę)
- Silniki serwo
- Deski prototypowe
- Przewody połączeniowe
- Kątomierze
- Władcy
- Kable USB do programowania
- Komputery z zainstalowanym Arduino IDE
- Projektor lub tablica do demonstracji

Zajęcia

Dzień 1

<u>Wprowadzenie do trygonometrii (15 minut):</u>

- Rozpocznij lekcję od wprowadzenia pojęcia trygonometrii i jej znaczenia w zastosowaniach w świecie rzeczywistym.
- Krótko wyjaśnij sinus, cosinus i tangens jako stosunki boków w trójkątach prostokątnych.
- Omów, w jaki sposób funkcje trygonometryczne można wykorzystać do rozwiązywania problemów związanych z kątami i odległościami.

<u>Podstawy trygonometrii (30 minut):</u>

- Zanurz się głębiej w definicje sinusa, cosinusa i tangensa.
- Podaj przykłady wykorzystania tych funkcji do znajdowania brakujących kątów lub długości boków w trójkątach prostokątnych.
- Pozwól uczniom ćwiczyć rozwiązywanie podstawowych problemów trygonometrycznych na papierze.

<u>Wprowadzenie do Arduino i serwomotorów (20 minut):</u>

- Przedstaw Arduino jako platformę do tworzenia gadżetów i wyjaśnij jej elementy składowe (mikrokontroler, czujniki, elementy wykonawcze).
- Wyjaśnij koncepcję serwomotorów i sposobu sterowania nimi, aby poruszały się pod określonymi kątami.
- Pokaż przykłady sterowania serwomotorem za pomocą Arduino.

<u>Zajęcia praktyczne (45 minut):</u>

- Podziel uczniów na pary lub małe grupy.
- Zapewnij każdej grupie płytkę Arduino, serwosilnik, płytkę stykową i przewody połączeniowe.
- Poleć uczniom, aby złożyli proste urządzenie do pomiaru kąta serwomotoru, korzystając z kątomierza.
- Poinstruuj uczniów, jak pisać kod Arduino, aby sterować serwomotorem tak, aby poruszał się pod określonym kątem w oparciu o dane wprowadzone przez użytkownika.

Dzień 2

Przegląd koncepcji trygonometrii (20 minut):

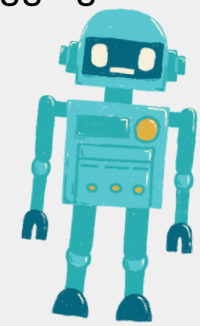
- Rozpocznij drugi Dzień od przeglądu pojęć sinus, cosinus i tangens poznanych Dniu 1.
- Zapewnij uczniom dodatkowe problemy praktyczne do rozwiązania za pomocą funkcji trygonometrycznych.

Programowanie urządzenia do pomiaru kąta (45 minut):

- Kontynuuj ćwiczenia praktyczne z dnia 1.
- Poinstruuj uczniów, aby uzupełnili kod Arduino w celu dokładnego pomiaru i wyświetlenia kątów za pomocą serwomotoru i wyświetlacza numerycznego (np. monitora szeregowego).
- Zachęć uczniów, aby zastosowali trygonometrię do przeliczenia położenia serwa na kąty.

Oto przykładowy kod Arduino umożliwiający utworzenie prostego urządzenia do pomiaru kąta za pomocą serwosilnika. Ten kod pozwala serwomotorowi przesunąć się o określony kąt w oparciu o dane wprowadzone przez użytkownika i wyświetla zmierzony kąt na monitorze szeregowym.

```
#include <Servo.h>
Servo myservo; // Create a Servo object
void setup() {
myservo.attach(9); // Attaches the servo to digital pin 9
Serial.begin(9600); // Initialize serial communication for debugging
}
void loop() {
int angle; // Variable to store the desired angle
Serial.println("Enter an angle (0-180): ");
while (!Serial.available()) {
  // Wait for user input
}
angle = Serial.parseInt(); // Read the angle entered by the user
if (angle >= 0 && angle <= 180) {
// Check if the entered angle is within the valid range
myservo.write(angle); // Move the servo to the specified angle
delay(500); // Delay for stability
Serial.print("Measured angle: ");
Serial.print(angle);
Serial.println(" degrees");
} else {
Serial.println("Invalid angle. Please enter a value between 0 and 180.");
}
}
```



W tym kodzie:

1. Dołączamy bibliotekę Servo do sterowania silnikiem serwo.
2. W funkcji setup() przyłączamy silnik serwo do cyfrowego pinu 9 i inicjalizujemy komunikację szeregową w celu debugowania.
3. W funkcji loop() odczytujemy wprowadzone przez użytkownika żądane kąty za pomocą Monitora Szeregowego. Użytkownik zostaje poproszony o podanie kąta między 0 a 180 stopni.
4. Sprawdzamy, czy wprowadzony kąt mieści się w prawidłowym zakresie (0-180 stopni). Jeśli tak, przesuwamy silnik serwo do określonego kąta za pomocą myservo.write(kąt) i wyświetlamy zmierzony kąt na Monitorze Szeregowym.
5. Jeśli wprowadzony kąt znajduje się poza prawidłowym zakresem, wyświetlamy komunikat o błędzie.
6. Aby użyć tego kodu z uczniami, upewnij się, że podłączą silnik serwo do cyfrowego pinu 9 na Arduino i otworzą Monitor Szeregowy, aby wprowadzać kąty i obserwować zmierzone kąty.

Dzień 3

Prezentacja Projektu i Dyskusja (60 minut):

- Każda grupa prezentuje swoje urządzenie do pomiaru kątów na Arduino przed klasą.
- Zachęcamy uczniów do wyjaśnienia, jak zastosowali koncepcje trygonometrii w swoich projektach.
- Omówienie realnych zastosowań urządzeń do pomiaru kątów i trygonometrii.
- Prowadzenie dyskusji w klasie na temat wyzwań i rozwiązań napotkanych podczas projektu.

Oceny

Oceniaj uczniów na podstawie ich udziału, jakości kodu, dokładności pomiaru kąta i umiejętności wyjaśniania zasad trygonometrycznych leżących u podstaw ich gadżetu.

Praca domowa

Przypisz zadanie Praca domowa, które wymaga od uczniów zbadania i zaprezentowania rzeczywistego zastosowania trygonometrii w różnych dziedzinach, takich jak inżynieria, fizyka czy architektura.

Wniosek

Zakończ lekcję, podsumowując kluczowe poznane koncepcje trygonometryczne i podkreślając ich praktyczne zastosowanie w rozwoju gadżetów Arduino. Podkreśl związek matematyki z technologią.

FROM GAME TO STEM

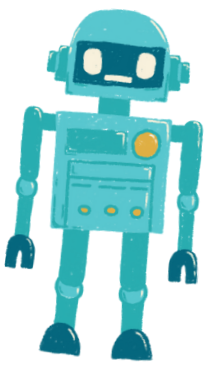
With the support of the Erasmus+ Programme of the European Union

Lekcja 2
Lekcja matematyki
Arduino

Modelowanie algebraiczne za pomocą Arduino

Czas trwania:

2 okresy



Cel:

- Studenci dowiedzą się, jak używać pojęć algebraicznych do rozwiązywania problemów świata rzeczywistego.
- Studenci wykorzystają umiejętności modelowania matematycznego i programowania, aby stworzyć gadżet Arduino, który rozwiązuje praktyczny problem.
- Studenci zdobędą doświadczenie ze zmiennymi i równaniami.

Materiały:

- Tablice Arduino (po jednej na ucznia lub grupę)
- Deski prototypowe
- Czujniki lub urządzenia wejściowe (np. czujnik temperatury, czuj światła, przycisk)
- Diody LED, rezystory i przewody połączeniowe
- Kable USB do programowania
- Komputery z zainstalowanym Arduino IDE
- Projektor lub tablica do demonstracji

Zajęcia

Dzień 1

Wprowadzenie do Modelowania Algebraicznego (15 minut):

- Rozpocznij lekcję od wprowadzenia do modelowania algebraicznego i jego znaczenia w rozwiązywaniu problemów rzeczywistych.
- Omów znaczenie zmiennych i równań w matematycznym modelowaniu.

Zmienne i Równania (30 minut):

- Przegląd pojęcia zmiennych i sposobu ich wykorzystywania do reprezentowania nieznanych wartości.
- Wprowadź równania liniowe (np. y=mx+b) jako sposób modelowania związków między zmiennymi.
- Przedstaw przykłady problemów z życia codziennego, które można reprezentować za pomocą równań.

Podstawy Arduino (20 minut):

- Wprowadź Arduino jako platformę do tworzenia gadżetów i wyjaśnij jej komponenty.
- Pokaż przykłady prostych projektów Arduino i sposób wykorzystania zmiennych w programowaniu.
-

Aktywność Praktyczna (45 minut):

- Podziel uczniów na pary lub małe grupy.
- Daj każdej grupie płytę Arduino, płytę prototypową, czujnik lub urządzenie wejściowe (np. czujnik temperatury) oraz diodę LED.
- Instruuj uczniów, aby stworzyli prosty projekt Arduino, który wykorzystuje czujnik do odczytu danych i diodę LED do wizualnego przedstawienia tych danych.
- Zachęć uczniów do wykorzystania zmiennych do przechowywania odczytów czujnika i tworzenia równań do kontrolowania diody LED na podstawie danych z czujnika.
- Omów różne scenariusze z życia codziennego, w których ich gadżet mógłby być przydatny.

Dzień 2

Przegląd pojęć algebraicznych (20 minut):

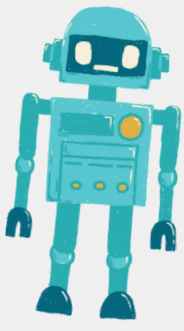
- Rozpocznij drugi Dzień od przeglądu pojęć zmiennych, równań i modelowania matematycznego.
- Omów projekty Arduino z poprzedniego dnia i ich zastosowania.

Programowanie modelu algebraicznego (45 minut):

- Kontynuuj ćwiczenia praktyczne z dnia 1.
- Poinstruuj uczniów, aby zmodyfikowali kod Arduino w celu utworzenia modelu matematycznego na podstawie danych z czujników.
- Zachęć ich do eksperymentowania z różnymi równaniami i zmiennymi w celu przedstawienia związku między danymi czujnika a sygnałem wyjściowym diody LED.
- Omów, w jaki sposób można wykorzystać model do prognozowania lub sterowania systemami w świecie rzeczywistym.

Oto przykład kodu Arduino dla prostego projektu modelowania algebraicznego. W tym projekcie uczniowie użyją czujnika temperatury do odczytu danych o temperaturze i sterowania diodą LED na podstawie odczytu temperatury. Kod wykorzystuje zmienne i równanie, aby określić, kiedy dioda LED powinna się włączyć, a kiedy wyłączyć.

```
// Define the pins for the temperature sensor, LED, and a resistor (if needed)
const int temperatureSensorPin = A0; // Analog pin for the temperature sensor
const int ledPin = 13; // Use the built-in LED on most Arduino boards
const float thresholdTemperature = 25.0; // Define the threshold temperature
void setup() {
pinMode(ledPin, OUTPUT); // Set the LED pin as an output
Serial.begin(9600); // Initialize serial communication for debugging (optional)
}
void loop() {
// Read the temperature from the sensor
int sensorValue = analogRead(temperatureSensorPin);
// Convert the analog value to temperature in degrees Celsius
float temperatureCelsius = map(sensorValue, 0, 1023, 0, 100); // Adjust the mapping as
needed
// Check if the temperature is above the threshold
if (temperatureCelsius > thresholdTemperature) {
digitalWrite(ledPin, HIGH); // Turn the LED on
} else {
digitalWrite(ledPin, LOW); // Turn the LED off
}
// Print the temperature to the serial monitor for debugging (optional)
Serial.print("Temperature: ");
Serial.print(temperatureCelsius);
Serial.println(" °C");
// Delay for a moment to avoid rapid LED toggling
delay(1000); // Delay for 1 second
}
```



W tym kodzie:

Definiujemy piny czujnika temperatury (podłączonego do pinu analogowego), diody LED (zwykle wbudowanej diody LED na większości płytek Arduino) oraz próg temperatury, który określa, kiedy dioda LED powinna się włączyć.
W funkcji setup() ustawiamy pin LED jako wyjście i inicjujemy komunikację szeregową w celu debugowania (opcjonalnie).
W funkcji pętli() odczytujemy temperaturę z czujnika za pomocą analogRead(). Wartość czujnika analogowego przeliczamy na temperaturę w stopniach Celsjusza w oparciu o charakterystykę czujnika.
Następnie sprawdzamy, czy temperatura przekracza zdefiniowany próg (thresholdTemperature). Jeżeli temperatura jest powyżej progu, dioda LED zostaje włączona; w przeciwnym razie jest wyłączony.
Opcjonalnie: Drukujemy temperaturę na monitorze szeregowym w celu debugowania.
Aby użyć tego kodu, uczniowie będą musieli podłączyć czujnik temperatury (np. LM35) do styku analogowego Arduino i diodę LED do styku cyfrowego. Kod odczyta temperaturę z czujnika i steruje diodą LED na podstawie odczytu temperatury i wartości progu.

Prezentacja projektu i dyskusja (30 minut):

- Każda grupa prezentuje klasie swój gadżet Arduino.
- Zachęć uczniów, aby wyjaśnili stworzony przez siebie model matematyczny i sposób jego działania.
- Ułatwienie dyskusji na zajęciach na temat zastosowań modelowania algebraicznego w rozwiązywaniu problemów praktycznych.

Oceny

Oceń uczniów na podstawie ich udziału, funkcjonalności gadżetu Arduino i ich umiejętności wyjaśniania pojęć algebraicznych zastosowanych w ich projekcie.

Praca domowa

Przypisz zadanie Praca domowa, które wymaga od uczniów zbadania i przedstawienia rzeczywistego problemu, który można rozwiązać za pomocą modelowania algebraicznego i gadżetu Arduino.

Wniosek

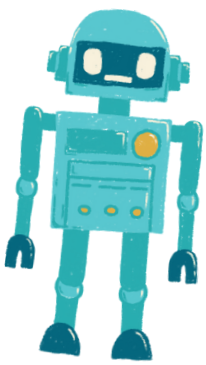
Zakończ lekcję podsumowaniem kluczowych poznanych pojęć algebraicznych i podkreśleniem znaczenia modelowania matematycznego w technologii i inżynierii.

Co-funded by the
Erasmus+ Programme
of the European Union

Lekcja 3
Lekcja fizyki Arduino

Odkrywanie ruchu i przyspieszenia za pomocą Arduino

Czas trwania:
3 okresy



Cel:

- Studenci poznają podstawowe zasady ruchu i przyspieszenia.
- Studenci będą stosować równania kinematyczne do rozwiązywania rzeczywistych problemów związanych z ruchem.
- Uczestnicy kursu zaprogramują Arduino do pomiaru i wyświetlania przyspieszenia za pomocą czujnika.

Materiały:

- Tablice Arduino (po jednej na ucznia lub grupę)
- Moduły czujników akcelerometru (np. ADXL345)
- Deski prototypowe
- Przewody połączeniowe
- Kable USB do programowania
- Komputery z zainstalowanym Arduino IDE
- Projektor lub tablica do demonstracji
- Opcjonalnie: Rzeczywiste obiekty do eksperymentów ruchowych (np. samochodziki, piłki)

Zajęcia

Dzień 1

<u>Wprowadzenie do ruchu i przyspieszenia (15 minut):</u>

- Rozpocznij lekcję od wprowadzenia pojęć ruchu i przyspieszenia.
- Zdefiniuj kluczowe pojęcia, takie jak prędkość, przyspieszenie i opóźnienie.
- Omów znaczenie zrozumienia ruchu w różnych dziedzinach, takich jak fizyka, inżynieria i transport.

<u>Fizyka Ruchu (30 minut):</u>

- Wyjaśnij równania ruchu, w tym:
-
- Przemieszczeni $s = ut + \frac{1}{2}at^2$
- Prędkość: $v = u + at$
-
- Przyśpieszenie: $a = \frac{v-u}{t}$
- Podaj przykłady wykorzystania tych równań do analizy ruchu.
- Przeprowadź proste eksperymenty ruchowe (np. toczenie piłki po zboczu), aby zademonstrować zasady ruchu.

<u>Wprowadzenie do Arduino i akcelerometrów (20 minut):</u>

- Przedstaw Arduino jako platformę do tworzenia gadżetów i wyjaśnij jej elementy.
- Wyjaśnij pojęcie akcelerometry i sposób, w jaki mierzą przyspieszenie.
- Pokaż przykłady danych wyjściowych akcelerometru i wyjaśnij, jaki mają one związek z ruchem w świecie rzeczywistym.

<u>Zajęcia praktyczne (45 minut):</u>

- Podziel uczniów na pary lub małe grupy.
- Zapewnij każdej grupie płytkę Arduino, moduł czujnika akcelerometru, płytkę stykową i przewody połączeniowe.
- Poleć uczniom, aby złożyli prosty obwód łączący akcelerometr z Arduino.
- Poinstruuj uczniów, jak pisać kod Arduino, aby odczytywać i wyświetlać dane dotyczące przyspieszenia z czujnika na monitorze szeregowym.

Dzień 2

Przegląd równań kinematycznych (20 minut):

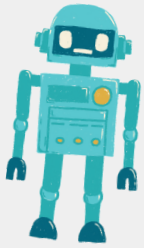
- Rozpocznij drugi Dzień od przejrzenia równań kinematycznych omówionych w Dniu 1.
- Podaj uczniom dodatkowe przykłady do rozwiązania za pomocą tych równań.

Programowanie urządzenia do pomiaru przyspieszenia (45 minut):

- Kontynuuj ćwiczenia praktyczne z dnia 1.
- Poinstruuj uczniów, aby uzupełnili kod Arduino w celu odczytania i wyświetlenia w czasie rzeczywistym danych dotyczących przyspieszenia z akcelerometru.
- Zachęć uczniów, aby w razie potrzeby skalibrowali czujnik i zastosowali równania przyspieszenia do interpretacji danych.

Oto przykład kodu Arduino do pomiaru i wyświetlania przyspieszenia za pomocą czujnika akcelerometru ADXL345. Ten kod umożliwi uczniom połączenie czujnika akcelerometru z Arduino i wyświetlenie wartości przyspieszenia na monitorze szeregowym.

```
#include <Wire.h>
#include <Adafruit_Sensor.h>
#include <Adafruit_ADXL345_U.h>
Adafruit_ADXL345_Unified accel = Adafruit_ADXL345_Unified(12345);
void setup(void) {
 Serial.begin(9600);
 Serial.println("Accelerometer Test"); Serial.println("");
  if(!accel.begin())
 {
/* There was a problem detecting the ADXL345 ... check your connections*/
   Serial.println("Ooops, no ADXL345 detected ... Check your wiring!");
   while(1);
 }
}
void loop(void) {
 sensors_event_t event;
 accel.getEvent(&event);
 /* Display the acceleration values (in m/s^2) */
 Serial.print("X: "); Serial.print(event.acceleration.x); Serial.print(" ");
 Serial.print("Y: "); Serial.print(event.acceleration.y); Serial.print(" ");
 Serial.print("Z: "); Serial.print(event.acceleration.z); Serial.println(" m/s^2");
 delay(500); // Delay for half a second between readings
}
```



W tym kodzie:
Dołączamy niezbędne biblioteki dla czujnika akcelerometru ADXL345.
Tworzymy obiekt Adafruit_ADXL345_Unified o nazwie accel, który będzie współpracował z czujnikiem.
W funkcji setup() inicjujemy komunikację szeregową w celu debugowania i sprawdzamy, czy wykryto czujnik akcelerometru. Jeżeli nie, wyświetli się komunikat o błędzie.
W funkcji pętli() stale odczytujemy dane przyspieszenia z czujnika za pomocą accel.getEvent(&event) i wyświetlamy wartości przyspieszenia osi X, Y i Z w metrach na sekundę do kwadratu (m/s^2) na Monitor szeregowy.
Aby użyć tego kodu, upewnij się, że uczniowie prawidłowo podłączyli czujnik akcelerometru ADXL345 do Arduino. Czujnik należy podłączyć do odpowiednich pinów (np. SDA i SCL) w celu komunikacji I2C. Kiedy Arduino jest włączone i uruchamia ten kod, będzie stale wyświetlać dane dotyczące przyspieszenia na monitorze szeregowym.
Należy pamiętać, że w celu użycia tego kodu może być konieczne zainstalowanie biblioteki Adafruit ADXL345 za pośrednictwem Menedżera bibliotek Arduino.

Prezentacja projektu i dyskusja (60 minut):

- Każda grupa prezentuje klasie swoje urządzenie do pomiaru przyspieszenia oparte na Arduino.
- Zachęć uczniów, aby wyjaśnili, w jaki sposób zastosowali równania kinematyczne i zasady fizyki w swoich projektach.
- Omów rzeczywiste zastosowania urządzeń do pomiaru przyspieszenia w różnych dziedzinach.
- Ułatwienie dyskusji w klasie na temat wyzwań i rozwiązań napotkanych podczas projektu.

Oceny

Oceniaj uczniów na podstawie ich udziału, jakości kodu, dokładności pomiaru przyspieszenia i umiejętności wyjaśniania zasad fizyki leżących u podstaw ich gadżetu.

Praca domowa

Przypisz zadanie Praca domowa, które wymaga od uczniów zbadania i zaprezentowania rzeczywistych zastosowań pomiaru przyspieszenia w fizyce lub inżynierii

Wniosek

Zakończ lekcję podsumowaniem kluczowych poznanych koncepcji fizyki i podkreśleniem ich praktycznego zastosowania w rozwoju gadżetów Arduino. Podkreśl związek pomiędzy fizyką i technologią.

Co-funded by the
Erasmus+ Programme
of the European Union

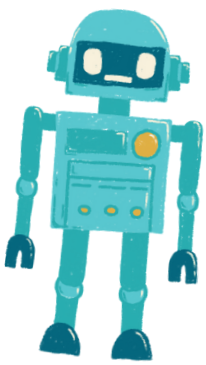
Lekcja 4

Lekcja fizyki Arduino

Odkrywanie oscylacji i ruchu wahadła za pomocą Arduino

Czas trwania:

2 okresy



Cel:

- Studenci będą rozumieć zasady ruchu oscylacyjnego i oscylacji harmonicznych.
- Studenci będą stosować modelowanie matematyczne do analizy ruchu wahadła.
- Uczestnicy kursu zaprogramują Arduino tak, aby symulował i wizualizował ruch wahadła.

Materiały:

- Tablice Arduino (po jednej na ucznia lub grupę)
- Silniki serwo
- Deski prototypowe
- Przewody połączeniowe
- Małe przedmioty jako obciążniki wahadłowe (np. podkładki)
- Sznurek lub nić do wahadeł
- Linijki lub miarka
- Kable USB do programowania
- Komputery z zainstalowanym Arduino IDE
- Projektor lub tablica do demonstracji

Zajęcia

Dzień 1

Wprowadzenie do Oscylacji i Ruchu Wahadłowego (15 minut):

- Rozpocznij lekcję od wprowadzenia koncepcji ruchu oscylacyjnego i jego znaczenia w różnych dziedzinach, w tym w fizyce i inżynierii.
- Zdefiniuj kluczowe pojęcia, takie jak okres, częstotliwość, amplituda i drgania harmoniczne.
- Wyjaśnij podstawy ruchu wahadłowego i jego matematyczną reprezentację.

Modelowanie Matematyczne Wahadeł (30 minut):

- Omów matematyczny model prostego wahadła, w tym równanie na okres wahadła:

$$T = 2\pi\sqrt{\frac{L}{g}}$$

gdzie T oznacza okres, L jest długością wahadła, a g to przyspieszenie ziemskie.

- Przedstaw przykłady wykorzystania równania do obliczania okresu wahadła.
- Przeprowadź proste obliczenia związane z ruchem wahadła.

Wprowadzenie do Arduino i Silników Serwo (20 minut):

- Przedstaw Arduino jako platformę do tworzenia gadżetów i wyjaśnij jego komponenty (mikrokontroler, sensory, aktuatory).
- Wyjaśnij pojęcie silników serwo i sposób ich wykorzystania do symulacji ruchu wahadła.
- Pokaż przykłady sterowania silnikiem serwo za pomocą Arduino.

Aktywność Praktyczna (45 minut):

- Podziel uczniów na pary lub małe grupy.
- Daj każdej grupie płytę Arduino, silnik serwo, płytę prototypową, przewody połączeniowe, mały przedmiot jako ciężarek wahadła i sznurek.
- Instruuj uczniów, aby złożyli prosty symulator wahadła, używając silnika serwo do kontrolowania ruchu wahadła.
- Pomóż uczniom w napisaniu kodu Arduino do tworzenia drgań harmonicznych poprzez zmianę pozycji serwa.

Dzień 2

Recenzja Ruchu Wahadłowego (20 minut):

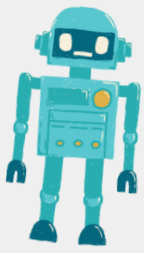
- Rozpocznij drugi dzień od powtórzenia koncepcji ruchu oscylacyjnego, ruch wahadłowego i matematycznego modelu prostego wahadła.
- Omów projekty Arduino z poprzedniego dnia i ich zastosowania.
-

Programowanie Symulatora Wahadła (45 minut):

- Kontynuuj aktywność praktyczną rozpoczętą pierwszego dnia.
- Instruuj uczniów, aby zmodyfikowali swój kod Arduino w taki sposób, aby dokładnie symulować ruch wahadła z różnymi parametrami, takimi jak długość i amplituda.
- Zachęć uczniów do wizualizacji i graficznej reprezentacji ruchu za pomocą silnika serwo.

Oto przykład kodu Arduino umożliwiającego utworzenie prostego symulatora wahadła z wykorzystaniem serwomotoru. Ten kod umożliwia uczniom zaprogramowanie Arduino w celu symulacji i wizualizacji ruchu wahadła:

```
#include <Servo.h>
Servo pendulum; // Create a Servo object
int angle = 90; // Initial angle of the pendulum (straight down)
int amplitude = 45; // Maximum angle to one side of the vertical (adjust as needed)
int period = 2000; // Period of the pendulum swing in milliseconds (adjust as needed)
void setup() {
  pendulum.attach(9); // Attach the servo to digital pin 9
}
void loop() {
  // Calculate the angle of the pendulum using harmonic motion
  int displacement = amplitude * cos(2 * PI * millis() / period);
  int pendulumAngle = angle + displacement;
   // Move the servo to the calculated angle
  pendulum.write(pendulumAngle);
  // Delay for a short time to control the speed of the simulation
  delay(50); // Adjust as needed for desired speed
}
```



W tym kodzie:
Dołączamy bibliotekę Servo do sterowania silnikiem serwo.
Tworzymy obiekt Servo zwany wahadłem, aby sterować silnikiem serwo.
Definiujemy zmienne dla początkowego kąta wahadła (kąt), maksymalnego kąta w jedną stronę od pionu (amplituda) i okresu wahadła (okres). Możesz dostosować te wartości, aby zmienić zachowanie wahadła.
W funkcji setup() podłączamy serwo do cyfrowego pinu 9.
W funkcji pętli() obliczamy kąt wahadła, korzystając ze wzoru na ruch harmoniczny. Przemieszczenie reprezentuje przemieszczenie kątowe od pozycji pionowej i dodajemy je do kąta początkowego, aby uzyskać Kąt wahadła.
Używamy pendulum.write(pendulumAngle), aby przesunąć serwo pod obliczony kąt.
Dodajemy opóźnienie, aby kontrolować prędkość symulacji. Dostosuj wartość opóźnienia według potrzeb, aby osiągnąć żądaną prędkość symulacji.
Aby użyć tego kodu, uczniowie powinni podłączyć serwomotor do cyfrowego styku 9 Arduino i przesłać kod na płytkę. Serwomotor będzie symulował ruch wahadła, a uczniowie będą mogli obserwować oscylacje w działaniu.

Prezentacja projektu i dyskusja (30 minut):

- Każda grupa prezentuje klasie swój symulator wahadła oparty na Arduino.
- Zachęć uczniów, aby wyjaśnili, w jaki sposób zastosowali zasady modelowania matematycznego w swoich projektach.
- Omów praktyczne zastosowania zrozumienia oscylacji harmonicznych w takich dziedzinach jak fizyka, inżynieria i astronomia.
- Ułatwienie dyskusji w klasie na temat wyzwań i rozwiązań napotkanych podczas projektu.

Oceny

Oceń uczniów na podstawie ich udziału, jakości gadżetu Arduino oraz ich umiejętności wyjaśniania zasad ruchu oscylacyjnego i oscylacji harmonicznych.

Praca domowa

Przypisz zadanie Praca domowa, które wymaga od uczniów zbadania i zaprezentowania rzeczywistego zastosowania oscylacji i ruchu harmonicznego w nauce lub technologii.

Wniosek

Zakończ lekcję podsumowaniem kluczowych poznanych koncepcji fizyki i podkreśleniem ich praktycznego zastosowania w rozwoju gadżetów Arduino. Podkreśl związek między matematyką i technologią w rozumieniu ruchu wahadła.

FROM GAME TO STEM

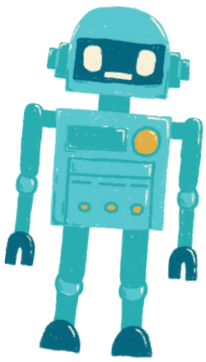
Co-funded by the Erasmus+ Programme of the European Union

Lekcja 5
Lekcja chemii Arduino

Odkrywanie hydroponiki i chemii wzrostu roślin

Czas trwania:

3 okresy



Cel:

- Studenci zrozumieją zasady hydroponiki i jej zalety we wzroście roślin.
- Studenci dowiedzą się, jakie składniki odżywcze są niezbędne do wzrostu roślin i jak przygotować roztwory odżywcze.
- Studenci zaprojektują i zbudują zautomatyzowany system nawadniania przy użyciu Arduino.
- Studenci będą monitorować i kontrolować poziom pH w systemie hydroponicznym, aby zoptymalizować wzrost roślin.

Materiały::

- Tablice Arduino (po jednej na ucznia lub grupę)
- Czujniki pH i roztwory do kalibracji pH
- Pompy wodne
- Emitery rurkowe i kroplowe
- Zbiornik na pożywkę
- Roztwory o pH podwyższającym obniżającym pH
- Roztwory buforowe pH
- Posadź nasiona lub sadzonki
- Podłoże hydroponiczne (np. wełna mineralna, perlit)
- Składniki pożywek (np. nawóz N-P-K)
- Pojemniki do instalacji hydroponicznych
- Przewody połączeniowe
- Kable USB do programowania
- Komputery z zainstalowanym Arduino IDE
- Projektor lub tablica do demonstracji

Zajęcia

Dzień 1

Wprowadzenie do hydroponiki (15 minut):

- Rozpocznij lekcję od przedstawienia koncepcji hydroponiki i jej zalet we wzroście roślin.
- Omów zalety kontrolowanych środowisk i systemów oszczędzających wodę.

Chemia wzrostu roślin (30 minut):

- Wyjaśnij pierwiastki chemiczne niezbędne do wzrostu roślin (np. azot, fosfor, potas) i ich rolę.
- Omów znaczenie poziomu pH dla pobierania składników odżywczych i zdrowia roślin.
- Wprowadzenie koncepcji roztworów odżywczych i kontroli pH w hydroponice.

Wprowadzenie do Arduino i czujników (20 minut):

- Przedstaw Arduino jako platformę do automatyzacji i gromadzenia danych oraz wyjaśnij jej elementy (mikrokontroler, czujniki).
- Pokaż przykłady czujników stosowanych w systemach hydroponicznych (np. czujniki pH).

Przygotowanie do ćwiczeń praktycznych (45 minut):

- Zapewnij uczniom płytki Arduino, czujniki pH, pompy wodne, rurki i pojemniki.
- Poleć uczniom, aby przeprowadzili burzę mózgów i zaplanowali projekt systemu hydroponicznego, uwzględniając roztwory odżywcze i kontrolę pH.

Dzień 2

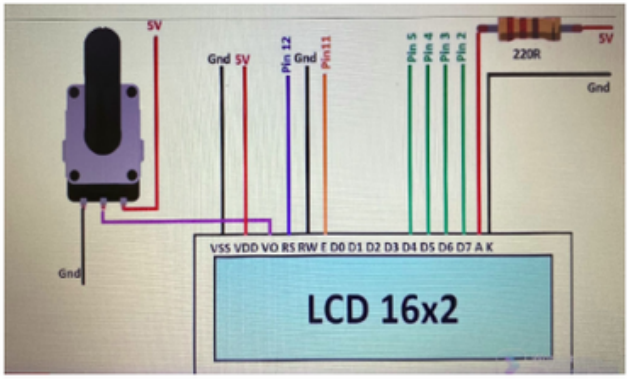
Konfiguracja systemu hydroponicznego (60 minut):

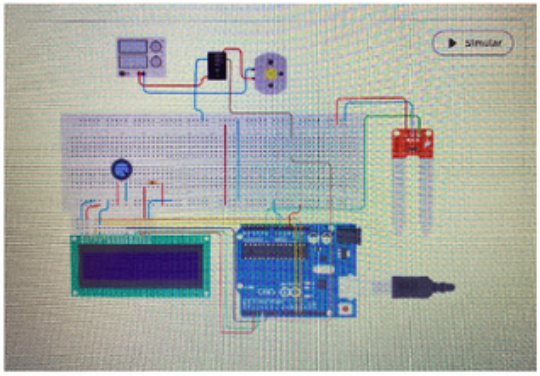
- Rozpocznij drugi Dzień, umożliwiając uczniom założenie systemów hydroponicznych.
- Uczniowie powinni zasadzić nasiona lub sadzonki w podłożu hydroponicznym (np. wełnie mineralnej) i wyposażyć system nawadniający w rurki i emitery kroplowe.
- Zademonstruj, jak mieszać i przygotowywać roztwory odżywcze.

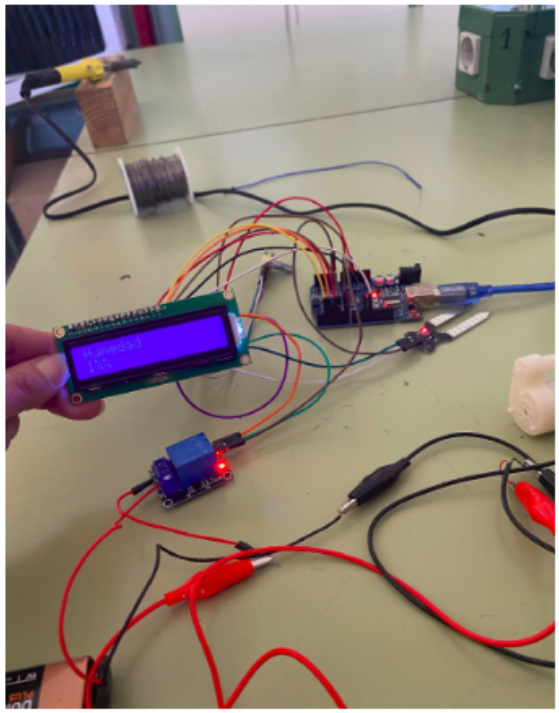
Monitorowanie i kalibracja pH (30 minut):

- Poinstruuj uczniów, jak kalibrować i używać czujników pH do monitorowania pH.
- Wyjaśnij znaczenie utrzymywania pH w optymalnym zakresie dla wchłaniania składników odżywczych (zwykle około pH 6-7).
- Poinstruuj uczniów, jak kalibrować czujniki pH przy użyciu roztworów buforowych pH.

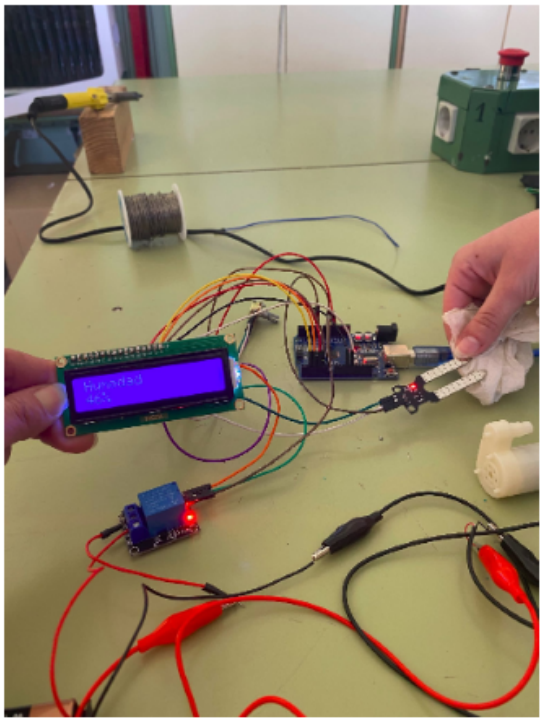
Schemat połączeń:







1% Wilgotność: Pompa działa



Wilgotność 46%: Pompa pracuje dalej

Dzień 3

Programowanie Arduino (60 minut):

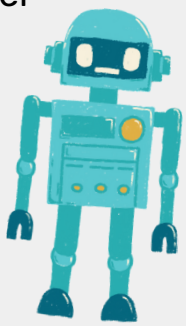
- Poleć uczniom, aby napisali kod Arduino dla automatycznego systemu nawadniania.
- Uczniowie powinni zaprogramować Arduino tak, aby sterował harmonogramem pompy wodnej i monitorował poziom pH.
- Podkreśl znaczenie regularnego podlewania i regulacji pH dla wzrostu roślin.

Zbieranie danych i kontrola pH (30 minut):

- Wyjaśnij, jak zbierać dane z czujników pH i monitorować poziomy pH.
- Omów działania, jakie uczniowie powinni podjąć, jeśli poziom pH odbiega od optymalnego zakresu.

Oto przykład kodu Arduino dla zautomatyzowanego systemu nawadniania hydroponicznego z monitorowaniem i kontrolą pH. Ten kod to podstawowa struktura, którą możesz dostosowywać i rozszerzać w zależności od konkretnej konfiguracji i potrzeb. Obejmuje monitorowanie pH, regulację pH i sterowanie pompą wodną do nawadniania.

```
#include <Adafruit_ADS1015.h>
#include <Wire.h>
#define PH_SENSOR_PIN 0 // Analog pin for pH sensor
#define PUMP_PIN 12    // Digital pin for water pump
// Create an Adafruit ADS1015 ADC object
Adafruit_ADS1015 ads;
float currentpH;
float targetpH = 6.5; // Adjust this value to your desired pH level
void setup() {
 Serial.begin(9600);
 ads.begin();
 pinMode(PUMP_PIN, OUTPUT);
 digitalWrite(PUMP_PIN, LOW); // Initialize pump as off
}
void loop() {
 // Read pH value from pH sensor
 currentpH = readpH();
 // Display current pH value
 Serial.print("Current pH: ");
 Serial.println(currentpH);
 // Check and adjust pH level
 if (currentpH < targetpH) {
   // pH is too low, add pH up solution (adjust as needed)
   // Implement pH adjustment mechanism here
     // For example, you can control a peristaltic pump for adding pH up
solution
```



```
digitalWrite(PUMP_PIN, HIGH); // Turn on the pump
 delay(1000); // Adjust the delay time as needed
 digitalWrite(PUMP_PIN, LOW); // Turn off the pump
 } else if (currentpH > targetpH) {
 // pH is too high, add pH down solution (adjust as needed)
 // Implement pH adjustment mechanism here
  // For example, you can control a peristaltic pump for adding pH down solution
 digitalWrite(PUMP_PIN, HIGH); // Turn on the pump
 delay(1000); // Adjust the delay time as needed
 digitalWrite(PUMP_PIN, LOW); // Turn off the pump
 }
 // Add code for irrigation control here (e.g., based on time intervals)
}
float readpH() {
 // Read pH value from the pH sensor and convert to pH scale
 int16_t rawValue = ads.readADC_SingleEnded(PH_SENSOR_PIN);
 float voltage = (rawValue * 0.1875) / 1000.0; // Convert to voltage
  float pHValue = 3.5 * voltage + 3.5; // Convert to pH scale (adjust calibration values as needed)
 return pHValue;
}
```

W tym kodzie:

Do współpracy z przetwornikiem ADS1015 ADC, który służy do odczytu wartości czujnika pH, wykorzystujemy bibliotekę Adafruit ADS1015. Upewnij się, że masz zainstalowaną tę bibliotekę w Arduino IDE.

Funkcja setup() inicjuje komunikację szeregową dla wyprowadzania danych, inicjuje przetwornik ADC i konfiguruje pin pompy wodnej jako wyjście.

W funkcji pętli() w sposób ciągły odczytujemy wartość pH z czujnika pH za pomocą funkcji readpH().

Porównujemy aktualną wartość pH z docelowym poziomem pH (targetpH) i w razie potrzeby dostosowujemy pH. Powinieneś wdrożyć mechanizm regulacji pH w oparciu o konkretną konfigurację, co może obejmować sterowanie pompą perystaltyczną w celu dodawania roztworów o pH podwyższającym lub obniżającym.

Dodatkowo możesz dodać kod sterujący pompą wody do nawadniania na podstawie przedziałów czasowych lub innych kryteriów.

Należy pamiętać, że ten kod zapewnia podstawowe ramy i może być konieczne jego skalibrowanie i dostrojenie zgodnie z konkretną konfiguracją czujnika i pompy, a także pożądanymi poziomami pH i harmonogramem nawadniania. Podczas pracy z chemikaliami i pompami należy przestrzegać środków bezpieczeństwa.

Dzień 4

Przygotowanie i monitorowanie eksperymentu (60 minut):

- Pozwól uczniom skonfigurować zautomatyzowane systemy hydroponiczne w szklarni lub klasie.
- Uczniowie powinni uruchomić system i monitorować wzrost roślin, poziom składników odżywczych i pH.

Prezentacja projektu i dyskusja (60 minut):

- Każda grupa przedstawia klasie swój projekt systemu hydroponicznego, metodologię i wstępne wyniki.
- Omów wpływ roztworów odżywczych i kontroli pH na wzrost roślin.
- Zachęć uczniów, aby na podstawie wstępnych obserwacji zaproponowali optymalizacje swoich systemów.

Oceny

Oceniaj uczniów na podstawie ich udziału, konfiguracji systemu, kalibracji pH, gromadzenia danych, analizy i jakości ich prezentacji.

Praca domowa

Przypisz pracę domową, która wymaga od studentów zbadania i zaprezentowania rzeczywistego zastosowania hydroponiki w rolnictwie lub rolnictwie zrównoważonym.

Wniosek

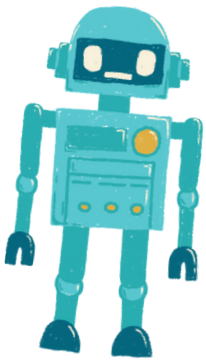
Zakończ lekcję, podsumowując kluczowe poznane koncepcje chemiczne i podkreślając znaczenie hydroponiki w zrównoważonym rolnictwie i produkcji żywności. Podkreśl rolę technologii (Arduino) w automatyzacji i optymalizacji systemów hydroponicznych.

Lekcja 6
Lekcja chemii Arduino

Badanie reakcji chemicznych za pomocą Arduino

Czas trwania:

3 okresy



Cel:

- Studenci będą rozumieć zasady reakcji chemicznych i ich znaczenie w życiu codziennym.
- Studenci będą projektować i przeprowadzać eksperymenty mające na celu obserwację i pomiar reakcji chemicznych.
- Studenci zaprogramują system monitorowania temperatury oparty na Arduino w celu gromadzenia i analizowania danych z reakcji chemicznych.

Materiały::

- Tablice Arduino (po jednej na ucznia lub grupę)
- Czujniki temperatury (np. DS18B20 lub LM35)
- Odczynniki chemiczne do eksperymentów (np. soda oczyszczona, ocet)
- Pojemniki reakcyjne (np. zlew, probówki)
- Przewody połączeniowe
- Kable USB do programowania
- Komputery z zainstalowanym Arduino IDE
- Projektor lub tablica do demonstracji
- Okulary ochronne i fartuchy laboratoryjne

Zajęcia

Dzień 1

<u>Wprowadzenie do reakcji chemicznych (15 minut):</u>

- Rozpocznij lekcję od przedstawienia pojęcia reakcji chemicznych i ich roli w życiu każdego dnia.
- Omów znaczenie reakcji chemicznych w różnych dziedzinach, w tym w chemii, biologii i przemyśle.

<u>Podstawy reakcji chemicznych (30 minut):</u>

- Zanurz się w podstawy reakcji chemicznych, w tym reagentów, produktów i zachowania masy.
- Podaj przykłady typowych reakcji chemicznych i ich zastosowań.
- Należy podkreślać protokoły bezpieczeństwa podczas obchodzenia się z chemikaliami w eksperymentach.

<u>Wprowadzenie do Arduino i czujników (20 minut):</u>

- Przedstaw Arduino jako platformę do gromadzenia danych i wyjaśnij jej elementy (mikrokontroler, czujniki).
- Pokaż przykłady czujników temperatury i sposób ich podłączenia do Arduino w celu gromadzenia danych.

<u>Przygotowanie do ćwiczeń praktycznych (45 minut):</u>

- Zapewnij uczniom płytki Arduino, czujniki temperatury, chemikalia i pojemniki reakcyjne.
- Poproś uczniów, aby przeprowadzili burzę mózgów i zaplanowali eksperymenty związane z reakcjami chemicznymi, koncentrując się na zmianach temperatury.
- Omów środki ostrożności i zasady pracy w laboratorium.

Dzień 2

Przygotowanie eksperymentu i zbieranie danych (60 minut):

- Rozpocznij drugi Dzień, pozwalając uczniom na przygotowanie eksperymentów w zakresie reakcji chemicznych.
- Uczniowie powinni wymieszać wybrane chemikalia w pojemnikach reakcyjnych i ustawić czujniki temperatury.
- Poleć uczniom, aby napisali kod Arduino do monitorowania temperatury i gromadzenia danych.

Analiza danych i dyskusja (30 minut):

- Pomóż uczniom analizować dane i szukać zmian temperatury podczas reakcji chemicznych.
- Omów znaczenie zmian temperatury w reakcjach chemicznych oraz pojęcia reakcji egzotermicznej i endotermicznej.
- Zachęć uczniów do rysowania produktów na podstawie swoich odkryć.

Dzień 3

Programowanie systemu monitorującego (60 minut):

- Poinstruuj uczniów, aby dostroili kod Arduino na potrzeby gromadzenia i analizy danych.
- Uczestnicy kursu powinni zaprogramować Arduino tak, aby rejestrował dane dotyczące temperatury w określonych odstępach czasu i wyświetlał je graficznie (np. na ekranie LCD lub monitorze szeregowym).

Prezentacja projektu i dyskusja (60 minut):

- Każda grupa przedstawia klasie konfigurację eksperymentu reakcji chemicznej, metodologię i wyniki.
- Zachęć uczniów, aby wyjaśnili swoje obserwacje i konsekwencje zmian temperatury w reakcjach chemicznych.
- Ułatwienie dyskusji w klasie na temat rzeczywistych zastosowań reakcji chemicznych w różnych dziedzinach.

Oto przykład kodu Arduino, którego można użyć do monitorowania temperatury podczas eksperymentu reakcji chemicznej. Ten kod odczytuje dane temperatury z czujnika temperatury DS18B20 i wyświetla je na monitorze szeregowym. Możesz dostosować i rozszerzyć ten kod, aby dopasować go do konkretnego eksperymentu.

```
#include <OneWire.h>
#include <DallasTemperature.h>
// Data wire is plugged into digital pin 2
#define ONE_WIRE_BUS 2
// Create a OneWire object
OneWire oneWire(ONE_WIRE_BUS);
// Pass the OneWire reference to Dallas Temperature sensor
DallasTemperature sensors(&oneWire);
```

```
void setup() {
 // Initialize serial communication for data output
 Serial.begin(9600);

 // Start the temperature sensor library
 sensors.begin();
}

void loop() {
 // Request temperature readings
 sensors.requestTemperatures();

 // Read temperature in Celsius
 float temperatureC = sensors.getTempCByIndex(0);

 // Display temperature on the Serial Monitor
 Serial.print("Temperature: ");
 Serial.print(temperatureC);
 Serial.println(" °C");

 // Add code here for data storage or further analysis

 // Delay before the next temperature reading (adjust as needed)
 delay(1000); // 1-second delay
}
```

W tym kodzie:

Do połączenia z czujnikiem temperatury DS18B20 używamy biblioteki Dallas Temperatura. Upewnij się, że masz zainstalowaną tę bibliotekę w Arduino IDE.
Funkcja setup() inicjuje komunikację szeregową w celu wyprowadzenia danych i uruchamia bibliotekę czujników temperatury.

W funkcji pętli() program w sposób ciągły żąda i odczytuje dane o temperaturze z czujnika za pomocą Sensors.getTempCByIndex(0). Wartość 0 oznacza pierwszy (i w tym przypadku jedyny) podłączony czujnik temperatury.
Dane dotyczące temperatury są wyświetlane na monitorze szeregowym z 1 sekundowym opóźnieniem pomiędzy odczytami. Można dostosować czas opóźnienia, aby kontrolować częstotliwość gromadzenia danych.

Możesz zmodyfikować ten kod, aby uwzględnić dodatkowe czujniki do różnych eksperymentów, zaimplementować przechowywanie danych na karcie SD lub utworzyć graficzną reprezentację danych na ekranie LCD, w zależności od konkretnych wymagań.

Oceny

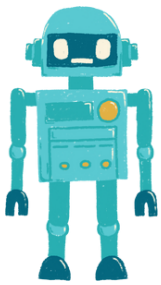
Oceniaj uczniów na podstawie ich udziału w eksperymencie, zbierania danych, analizy i jakości ich prezentacji.

Praca domowa

Assign Praca domowa that requires students to research and present a real-worlad application of chemical reactions in a specific industry or field of science.

Wniosek

Zakończ lekcję, podsumowując kluczowe poznane pojęcia chemiczne i podkreślając znaczenie reakcji chemicznych w zrozumieniu naturalnych procesów i postępu technologicznego. Podkreśl rolę technologii (Arduino) w badaniach naukowych.



FROM GAME TO STEM

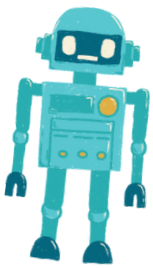
Co-funded by the Erasmus+ Programme of the European Union

Lekcja 7
Lekcja biologii Arduino

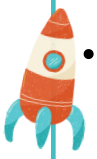
Badanie przewodności cieplnej za pomocą termometrów Arduino

Czas trwania:

3 okresy



Cel:



- Studenci rozumieją pojęcie przewodności cieplnej i jego znaczenie.
- Studenci zaprojektują i przeprowadzą doświadczenie mające na celu określenie przewodności cieplnej różnych materiałów.
- Uczniowie będą używać termometrów Arduino do gromadzenia danych o temperaturze i analizowania uzyskanych wyników.

Materiały:

- Tablice Arduino (po jednej na ucznia lub grupę)
- Czujniki temperatury (np. DS18B20 lub LM35)
- Deski prototypowe
- Przewody połączeniowe
- Różne materiały do badania przewodności (np. metale, tworzywa sztuczne, drewno, szkło)
- Materiały izolacyjne (np. pianka, tkanina)
- Gorąca woda lub źródło ciepła
- Zimna woda lub lód
- Stoper lub minutnik
- Kable USB do programowania
- Komputery z zainstalowanym Arduino IDE
- Projektor lub tablica do demonstracji
- Okulary i rękawice ochronne (do pracy z gorącymi materiałami)

Zajęcia

Dzień 1

<u>Wprowadzenie do przewodności cieplnej (15 minut):</u>

- Rozpocznij lekcję od przedstawienia pojęcia przewodności cieplnej i tego, dlaczego jest ona ważna w życiu każdego dnia.
- Omów rzeczywiste zastosowania przewodności cieplnej, takie jak gotowanie, materiały budowlane i inżynieria.

<u>Przenikanie ciepła i izolacja (30 minut):</u>

- Wyjaśnij różne metody wymiany ciepła (przewodnictwo, konwekcja, promieniowanie) i skoncentruj się na przewodzeniu, które jest istotne dla eksperymentu.
- Omów materiały izolacyjne i ich rolę w ograniczaniu przenikania ciepła.
- Podkreśl potrzebę kontrolowanego eksperymentu w celu pomiaru przewodności cieplnej.

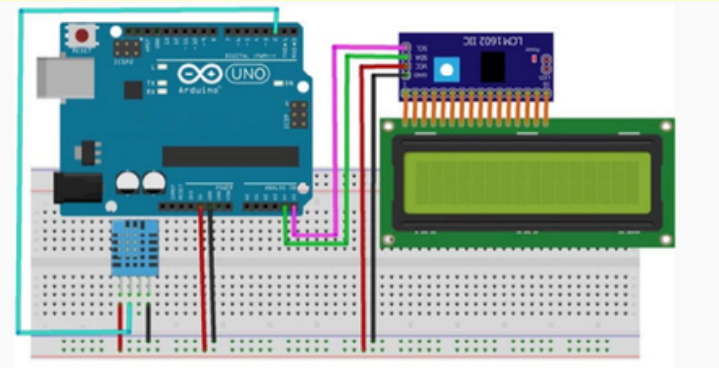
<u>Wprowadzenie do termometrów Arduino (20 minut):</u>

- Przedstaw Arduino jako platformę do gromadzenia danych i wyjaśnij jej elementy (mikrokontroler, czujniki).
- Wyjaśnij zastosowanie czujników temperatury i sposób ich podłączenia do płytek Arduino.
- Pokaż przykłady gromadzenia i wizualizacji danych o temperaturze za pomocą Arduino.

<u>Przygotowanie do ćwiczeń praktycznych (45 minut):</u>

- Zapewnij uczniom płytki Arduino, czujniki temperatury, płytki stykowe i przewody połączeniowe.
- Wyjaśnij układ eksperymentalny: Każda grupa przetestuje różne materiały, aby określić ich przewodność cieplną.
- Poleć uczniom, aby przeprowadzili burzę mózgów i zaplanowali swoje eksperymenty, w tym sposoby kontrolowania zmiennych i gromadzenia danych.

SZEMA:



Dzień 2

Przygotowanie eksperymentu i zbieranie danych (60 minut):

- Rozpocznij drugi Dzień, pozwalając uczniom na przygotowanie eksperymentów.
- Uczniowie powinni przymocować czujnik temperatury do wybranych materiałów oraz przygotować pojemniki z ciepłą i zimną wodą.
- Poleć uczniom, aby rozpoczęli zbieranie danych o temperaturze za pomocą Arduino i zapisali temperatury początkowe.
- Poproś uczniów, aby zmierzyli i zapisali czas potrzebny do zmiany temperatury w każdym materiale.

Analiza danych i dyskusja (30 minut):

- Pomóż uczniom analizować dane, obliczać szybkość zmian temperatury i porównywać przewodność różnych materiałów.
- Omów pojęcie oporu cieplnego i jego związek z przewodnością.
- Zachęć uczniów, aby na podstawie swoich ustaleń określili, który materiał jest najlepszym przewodnikiem ciepła, a który najgorszym.

Oto przykład kodu Arduino, który możesz wykorzystać w eksperymencie, aby zmierzyć i porównać przewodność cieplną różnych materiałów za pomocą czujnika temperatury (np. DS18B20). Kod ten zbierze dane o temperaturze z czujnika i umożliwi obliczenie szybkości zmian temperatury dla każdego materiału.

```
#include <OneWire.h>
#include <DallasTemperature.h>
// Data wire is plugged into digital pin 2
#define ONE_WIRE_BUS 2
// Create a OneWire object
OneWire oneWire(ONE_WIRE_BUS);
// Pass the OneWire reference to Dallas Temperature sensor
DallasTemperature sensors(&oneWire);
void setup() {
 // Initialize serial communication for data output
 Serial.begin(9600);
 // Start the temperature sensor library
 sensors.begin();
}
void loop() {
```

```
// Initialize variables for temperature measurements
 float initialTemp, finalTemp;
 unsigned long startTime, endTime;
 float rateOfChange;
 // Wait for the user to start the experiment (e.g., press a button)
while (!digitalRead(3)) {
  // Wait for button press
 }
 delay(1000); // Debounce delay
 // Measure initial temperature
 sensors.requestTemperatures(); // Request temperature readings
 initialTemp = sensors.getTempCByIndex(0); // Get temperature in Celsius
 // Record the start time
 startTime = millis();
 // Wait for the temperature to stabilize (e.g., 5 seconds)
 delay(5000);
 // Measure final temperature
 sensors.requestTemperatures();
 finalTemp = sensors.getTempCByIndex(0);
 // Record the end time
 endTime = millis();
 // Calculate the rate of temperature change (°C per second)
  rateOfChange = (finalTemp - initialTemp) / ((endTime - startTime) /
1000.0);
 // Output results to the Serial Monitor
 Serial.print("Initial Temperature: ");
 Serial.print(initialTemp);
 Serial.println(" °C");
 Serial.print("Final Temperature: ");
 Serial.print(finalTemp);
 Serial.println(" °C");
 Serial.print("Rate of Change: ");
 Serial.print(rateOfChange);
 Serial.println(" °C/s");
 // Wait for user input (e.g., press a button) to proceed to the next material
 Serial.println("Press a button to test the next material.");
 while (!digitalRead(3)) {
  // Wait for button press
 }
 delay(1000); // Debounce delay
}
```

W tym kodzie:
Do połączenia z czujnikiem temperatury DS18B20 używamy biblioteki Dallas Temperatura. Upewnij się, że masz zainstalowaną tę bibliotekę w Arduino IDE.
Definiujemy pin cyfrowy (np. pin 2), do którego podłączony jest przewód danych czujnika DS18B20.
W funkcji setup() inicjujemy komunikację szeregową w celu wyprowadzenia danych i uruchamiamy bibliotekę czujników temperatury.
W funkcji pętli() program czeka na naciśnięcie przycisku, aby rozpocząć eksperyment. Temperaturę mierzy się przed i po okresie oczekiwania (np. 5 sekund) w celu obliczenia szybkości zmian temperatury.
Szybkość zmian oblicza się jako różnicę temperatur podzieloną przez czas potrzebny do osiągnięcia tej zmiany. Daje to szybkość w °C na sekundę.
Wyniki są drukowane na monitorze szeregowym, łącznie z temperaturą początkową, temperaturą końcową i szybkością zmian.
Po zakończeniu eksperymentu dla jednego materiału możesz nacisnąć przycisk, aby przejść do następnego materiału.
Pamiętaj, aby prawidłowo podłączyć czujnik DS18B20 do Arduino i upewnić się, że przycisk jest podłączony do pinu cyfrowego (np. pinu 3) w celu uruchomienia eksperymentu. Dostosuj kod zgodnie z potrzebami w zależności od konkretnej konfiguracji.

Dzień 3

Prezentacja projektu i dyskusja (60 minut):

- Każda grupa przedstawia klasie plan eksperymentu, metodologię i wyniki.
- Zachęć uczniów, aby wyjaśnili swoje obserwacje, omówili potencjalne źródła błędów i zasugerowali ulepszenia.
- Ułatwiaj dyskusję w klasie na temat rzeczywistych implikacji ich ustaleń, takich jak wybór materiałów do izolacji termicznej lub materiałów przewodzących do radiatorów.

Oceny

Oceniaj uczniów na podstawie ich udziału w eksperymencie, zbierania danych, analizy i jakości ich prezentacji.

Praca domowa

Przypisz pracę domową, która wymaga od studentów zbadania i zaprezentowania rzeczywistego zastosowania przewodności cieplnej w inżynierii lub materiałoznawstwie.

Wniosek

Zakończ lekcję, podsumowując kluczowe poznane pojęcia i podkreślając znaczenie zrozumienia przewodności cieplnej w każdym życiu i zastosowaniach technologicznych.

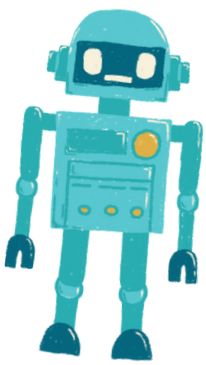
FROM GAME TO STEM

Co-funded by the Erasmus+ Programme of the European Union

Lekcja 8
Lekcja biologii Arduino

Odkrywanie fotosyntezy i wzrostu roślin za pomocą Arduino

Czas trwania:
3 okresy



Cel:

- Studenci poznają proces fotosyntezy i jego znaczenie dla wzrostu roślin.
- Studenci zaprojektują i przeprowadzą eksperyment mający na celu zbadanie wpływu czynników środowiskowych na fotosyntezę.
- Studenci zaprogramują system oparty na Arduino do monitorowania i gromadzenia danych związanych ze wzrostem roślin.

Materiały:

- Tablice Arduino (po jednej na ucznia lub grupę)
- Czujniki (np. czujnik światła, czujnik temperatury, czujnik wilgotności)
- Światła LED do uprawy (opcjonalnie)
- Rośliny doniczkowe lub nasiona
- Pojemniki na ziemię i sadzonki
- Przewody połączeniowe
- Kable USB do programowania
- Komputery z zainstalowanym Arduino IL
- Projektor lub tablica do demonstracji
- Doniczkowanie gleby, wody i innych materiałów do pielęgnacji roślin

Zajęcia

Dzień 1

<u>Wprowadzenie do fotosyntezy (15 minut):</u>

- Rozpocznij lekcję od wprowadzenia pojęcia fotosyntezy i jej znaczenia w rozwoju roślin.
- Omów chemiczne równanie fotosyntezy oraz rolę światła, dwutlenku węgla i wody w tym procesie.

<u>Czynniki środowiskowe i wzrost roślin (30 minut):</u>

- Wyjaśnij, jak różne czynniki środowiskowe, takie jak natężenie światła, temperatura i wilgotność, mogą wpływać na fotosyntezę i wzrost roślin.
- Przedyskutuj, dlaczego te czynniki są niezbędne dla zdrowego rozwoju roślin.
- Podkreśl potrzebę kontrolowanych eksperymentów w celu zbadania tych czynników.

<u>Wprowadzenie do Arduino i czujników (20 minut):</u>

- Przedstaw Arduino jako platformę do gromadzenia danych i wyjaśnij jej elementy (mikrokontroler, czujniki).
- Pokaż przykłady czujników powszechnie stosowanych w monitorowaniu środowiska i pielęgnacji roślin.
- Wyjaśnij rolę czujników w zbieraniu danych do eksperymentów.

<u>Przygotowanie do ćwiczeń praktycznych (45 minut):</u>

- Zapewnij uczniom płytki Arduino, czujniki (np. czujnik światła, czujnik temperatury, czujnik wilgotności) i lampy LED do uprawy (opcjonalnie).
- Poproś uczniów, aby przeprowadzili burzę mózgów i zaplanowali eksperymenty związane ze wzrostem roślin, koncentrując się na jednym czynniku środowiskowym.
- Niech uczniowie przygotują doniczki z ziemią i posadzą nasiona lub małe rośliny doniczkowe.

Dzień 2

Przygotowanie eksperymentu i zbieranie danych (60 minut):

- Rozpocznij drugi Dzień, pozwalając uczniom na przygotowanie eksperymentów.
- Uczniowie powinni umieścić czujniki w środowisku roślinnym, podłączyć je do Arduino i ustawić lampy LED do uprawy (jeśli są używane).
- Poleć uczniom, aby zebrali dane związane z wybranym czynnikiem środowiskowym, takim jak natężenie światła lub temperatura.
- Podkreśl znaczenie dokładnego i regularnego rejestrowania danych.

Analiza danych i dyskusja (30 minut):

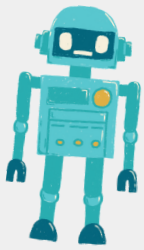
- Pomóc uczniom w analizie danych, poszukiwaniu trendów lub wzorców związanych ze wzrostem roślin i wybranym czynnikiem.
- Omów wpływ czynnika na fotosyntezę i rozwój roślin.
- Zachęć uczniów, aby czerpali wnioski ze swoich odkryć.

Dzień 3

Programowanie systemu monitorującego (60 minut):

- Poleć uczniom, aby napisali kod Arduino w celu monitorowania i zbierania danych z czujników.
- Uczniowie powinni zaprogramować Arduino tak, aby rejestrował dane w określonych odstępach czasu (np. co godzinę).

Oto przykład kodu Arduino dla prostego systemu monitorowania środowiska, który mierzy i rejestruje dane związane z natężeniem światła za pomocą czujnika światła. W razie potrzeby możesz dostosować ten kod do innych czynników środowiskowych:



```
#include <Wire.h>
#include <Adafruit_Sensor.h>
#include <Adafruit_TSL2591.h>
Adafruit_TSL2591 tsl = Adafruit_TSL2591(2591);
void setup() {
 Serial.begin(9600);
  // Initialize the light sensor
 if(!tsl.begin()) {
   Serial.println("Light sensor not found. Check wiring.");
   while(1);
 }
 tsl.setGain(TSL2591_GAIN_LOW);    // Adjust the gain (options: LOW, MED, HIGH, MAX)
 tsl.setTiming(TSL2591_INTEGRATIONTIME_100MS); // Adjust the integration time (options:
100, 200, 300, 400, 500, 600)
}
void loop() {
 // Read and print light intensity data
 uint16_t luminance = tsl.getLuminosity(TSL2591_VISIBLE);
 Serial.print("Light Intensity (Lux): ");
 Serial.println(luminance);
    // Add code here to log data to an SD card, display on an LCD, or transmit to a
computer/server.
 // Wait for a specific time interval (e.g., 1 hour)
 delay(3600000); // Adjust the delay time as needed
}
```

W tym kodzie:

Do współpracy z czujnikiem światła TSL2591 używamy biblioteki Adafruit TSL2591. Upewnij się, że masz zainstalowaną tę bibliotekę w Arduino IDE.
Funkcja setup() inicjuje komunikację szeregową w celu wyprowadzenia danych i konfiguruje czujnik światła. Konfiguruje również wzmocnienie i czas integracji czujnika w oparciu o Twoje wymagania.
W funkcji pętli() program w sposób ciągły odczytuje dane o natężeniu światła (w luksach) z czujnika za pomocą tsl.getLuminosity(TSL2591_VISIBLE).
Możesz dodać kod w pętli, aby zarejestrować dane na karcie SD, wyświetlić je na ekranie LCD lub przesłać do komputera lub serwera w celu dalszej analizy. Można na przykład użyć modułu karty SD do lokalnego przechowywania danych.
Opóźnienie na końcu pętli jest ustawione na oczekiwanie przez określony czas (np. 1 godzinę) przed wykonaniem następnego odczytu. Można dostosować czas opóźnienia, aby kontrolować częstotliwość gromadzenia danych.

Kod ten zapewnia podstawowe ramy monitorowania natężenia światła i można je rozszerzyć, dodając więcej czujników w celu monitorowania dodatkowych czynników środowiskowych, takich jak temperatura i wilgotność. Pamiętaj, aby dostosować kod do konkretnych czujników i sprzętu używanego w eksperymencie.

Prezentacja projektu i dyskusja (60 minut):

- Każda grupa przedstawia klasie konfigurację, metodologię i wyniki swojego eksperymentu ze wzrostem roślin.
- Zachęć uczniów, aby wyjaśnili swoje obserwacje i konsekwencje swoich odkryć dla pielęgnacji roślin i rolnictwa.
- Ułatwij dyskusję w klasie na temat znaczenia fotosyntezy w ekosystemie i tego, w jaki sposób technologia (Arduino) może pomóc w badaniach naukowych.

Oceny

Oceniaj uczniów na podstawie ich udziału w eksperymencie, zbierania danych, analizy i jakości ich prezentacji.

Praca domowa

Przypisz Praca domowa, która wymaga od studentów zbadania i zaprezentowania rzeczywistych zastosowań fotosyntezy i monitorowania środowiska w rolnictwie lub ekologii.

Wniosek

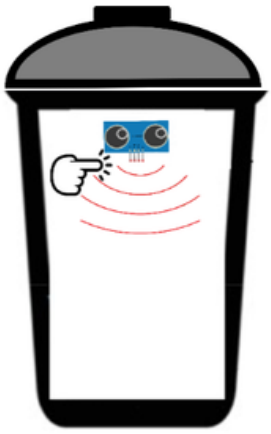
Zakończ lekcję, podsumowując kluczowe poznane pojęcia z biologii i podkreślając znaczenie fotosyntezy i czynników środowiskowych dla wzrostu roślin. Podkreśl rolę technologii w pogłębianiu wiedzy naukowej.

FROM GAME TO STEM

Co-funded by the Erasmus+ Programme of the European Union

Lekcja 9
Lekcja ekologii Arduino

Budowa inteligentnego kosza na śmieci za pomocą Arduino



Czas trwania:

3 okresy

Cel:

- Studenci poznają podstawy programowania Arduino i jego zastosowania w automatyce.
- Studenci dowiedzą się o czujnikach ultradźwiękowych i o tym, jak można je wykorzystać do wykrywania obiektów.
- Uczniowie zbudują prototyp inteligentnego pojemnika na śmieci i zaprogramują go tak, aby otwierał klapę po wykryciu obiektu.
- Studenci będą badać korzyści dla środowiska wynikające z inteligentnych systemów gospodarowania odpadami.

Materiały:

- Tablice Arduino Uno (po jednej na ucznia lub grupę)
- Czujnik ultradźwiękowy HC-SR04 (jeden na ucznia lub grupę)
- Serwomotory (jeden na ucznia lub grupę)
- Płytki stykowe i przewody połączeniowe
- Małe kartonowe pudełko lub pojemnik (na kosz na śmieci)
- Kartonowa lub plastikowa klapka (imitująca klapę kosza na śmieci)
- Kable USB do programowania
- Komputery z zainstalowanym Arduino IDE
- Projektor lub tablica do demonstracji

Zajęcia

Dzień 1

Wprowadzenie do Arduino i elektroniki (15 minut):

- Lekcję rozpocznij od przedstawienia Arduino jako platformy mikrokontrolera wykorzystywanej w różnych projektach.
- Omów znaczenie elektroniki i automatyki we współczesnej technologii.

Czujniki ultradźwiękowe i wykrywanie obiektów (30 minut):

- Wyjaśnij zasadę działania czujników ultradźwiękowych i sposób ich działania przy pomiarze odległości.
- Omów elementy czujnika ultradźwiękowego HC-SR04 (nadajnik i odbiornik).
- Zilustruj, w jaki sposób można wykorzystać czujniki ultradźwiękowe do wykrywania obiektów i pomiaru odległości.

Wprowadzenie do serwomotorów (20 minut):

- Przedstaw serwomotory i ich zastosowanie w sterowaniu ruchami mechanicznymi.
- Pokaż przykłady wykorzystania serwomotorów w projektach takich jak otwieranie klapy.

Przygotowanie do ćwiczeń praktycznych (45 minut):

- Zapewnij każdej grupie Arduino Uno, czujnik ultradźwiękowy, serwomotor, płytkę stykową i przewody połączeniowe.
- Poinstruuj uczniów, aby złożyli prototyp inteligentnego pojemnika na śmieci, odpowiednio umieszczając czujnik ultradźwiękowy i serwomotor.

Dzień 2

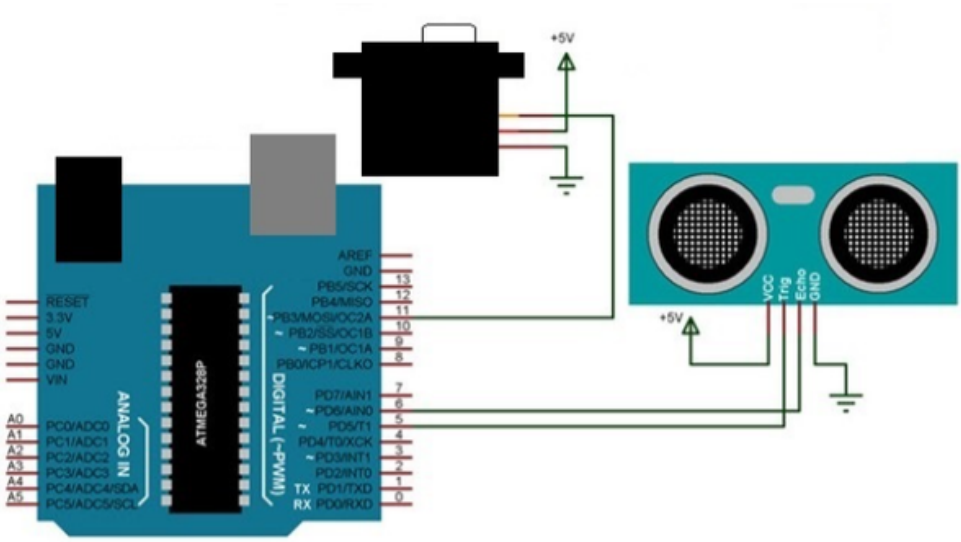
Podstawy programowania Arduino (30 minut):

- Naucz uczniów podstaw programowania Arduino, w tym setup(), Loop() i pinMode().
- Podaj przykłady podstawowego kodu Arduino do migania diody LED.

Programowanie inteligentnego kosza na śmieci (60 minut):

- Poinstruuj uczniów, jak pisać kod Arduino umożliwiający sterowanie inteligentnym koszem na śmieci w oparciu o odczyty czujnika ultradźwiękowego.
- Wyjaśnij logikę otwierania klapy w przypadku wykrycia obiektu w określonym zasięgu.

SZEMA:



Oto przykładowy kod Arduino dla inteligentnego pojemnika na śmieci wykorzystującego czujnik ultradźwiękowy (HC-SR04) i serwomotor. Kod ten umożliwia serwomotorowi otwarcie klapy pojemnika na śmieci w przypadku wykrycia obiektu w określonym zakresie:

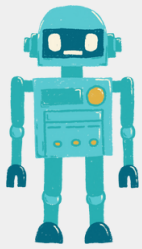
```
#include <Servo.h>

#define TRIGGER_PIN 9
#define ECHO_PIN 10
#define SERVO_PIN 11

Servo myservo; // Create a Servo object
int distance; // Variable to store distance measured by the Ultrasonic sensor

void setup() {
 myservo.attach(SERVO_PIN); // Attach the Servo to the specified pin
 pinMode(TRIGGER_PIN, OUTPUT);
 pinMode(ECHO_PIN, INPUT);
 Serial.begin(9600); // Initialize serial communication for debugging
}
```

```
void loop() {
 // Send a brief pulse to trigger the Ultrasonic sensor
 digitalWrite(TRIGGER_PIN, LOW);
 delayMicroseconds(2);
 digitalWrite(TRIGGER_PIN, HIGH);
 delayMicroseconds(10);
 digitalWrite(TRIGGER_PIN, LOW);
 // Read the duration of the echo pulse and calculate the distance
 duration = pulseIn(ECHO_PIN, HIGH);
 distance = (duration / 2) / 29.1; // Calculate distance in centimeters
 // Check if an object is within the specified range (adjust as needed)
 if (distance < 20) { // You can adjust the distance threshold here
  // If an object is detected, open the flap
   myservo.write(90); // Rotate the Servo to open the flap (adjust the angle as needed)
  delay(1000); // Wait for 1 second
  myservo.write(0); // Rotate the Servo back to close the flap
 }
 // Print the distance to the Serial Monitor for debugging
 Serial.print("Distance: ");
 Serial.print(distance);
 Serial.println(" cm");
 // Add a delay between readings to prevent rapid triggering
 delay(1000); // You can adjust the delay time as needed
}
```



W tym kodzie:
Dołączamy bibliotekę Servo i definiujemy numery pinów dla pinów wyzwalania i echa czujnika ultradźwiękowego, a także pinu sterującego serwomotoru.
W funkcji setup() podłączamy serwomotor do określonego pinu i konfigurujemy pin wyzwalający jako wyjście, a pin echo jako wejście. Inicjujemy również komunikację szeregową w celu debugowania.
Funkcja pętli() wielokrotnie wykonuje następujące kroki:
Wysyła krótki impuls do czujnika ultradźwiękowego w celu uruchomienia pomiaru odległości.
Mierzy czas trwania impulsu echa i oblicza odległość w centymetrach.
Sprawdza, czy jakiś obiekt znajduje się w określonym zasięgu (w tym przykładzie 20 cm) i jeśli tak, otwiera klapę pojemnika na śmieci, obracając serwomotor o zadany kąt (90 stopni).
Drukuje odległość do monitora szeregowego w celu debugowania.
Dodaje opóźnienie pomiędzy odczytami, aby zapobiec szybkiemu wyzwalaniu.
Możesz dostosować próg odległości, kąt serwomotoru i czasy opóźnienia, aby dopasować je do konkretnej konfiguracji i wymagań.

Testowanie i rozwiązywanie problemów (30 minut):

- Poproś uczniów, aby przetestowali prototypy inteligentnego pojemnika na śmieci i rozwiązali wszelkie problemy z kodem lub sprzętem.
- Zachęcaj do eksperymentowania z różnymi odległościami wykrywania i kątami serwosilnika.

Dzień 3

Prezentacja projektu i dyskusja (60 minut):

- Każda grupa prezentuje klasie swój projekt inteligentnego kosza na śmieci, wyjaśniając projekt, komponenty i kod.
- Omów korzyści dla środowiska wynikające z inteligentnych systemów gospodarki odpadami i ich potencjalny wpływ na redukcję ilości odpadów.

Otwarta dyskusja i przyszłe ulepszenia (30 minut):

- Ułatwiaj dyskusję w klasie na temat potencjalnych ulepszeń inteligentnego pojemnika na śmieci i innych zastosowań podobnej technologii.
- Zachęć uczniów do przeprowadzenia burzy mózgów na temat pomysłów na dalsze udoskonalanie rozwiązań w zakresie gospodarowania odpadami.

Oceny

Oceniaj uczniów na podstawie ich udziału, funkcjonalności projektu, zrozumienia programowania Arduino i ich umiejętności rozwiązywania problemów.

Praca domowa

Przypisz Praca domowa, która wymaga od studentów zbadania i przedstawienia rzeczywistych przykładów inteligentnych systemów gospodarki odpadami i ich wpływu na zrównoważony rozwój.

Wniosek

Zakończ lekcję, podsumowując kluczowe poznane koncepcje, podkreślając przecięcie technologii i rozwiązań środowiskowych oraz zachęcając uczniów do krytycznego myślenia na temat stosowania technologii w celu uzyskania pozytywnych zmian.

FROM GAME TO STEM

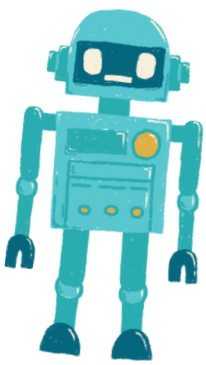
Co-funded by the Erasmus+ Programme of the European Union

Lekcja 10
Lekcja ekologii Arduino

Badanie jakości powietrza za pomocą Arduino

Czas trwania:

1 okresy



Cel:

- Studenci dowiedzą się, jak ważna jest jakość powietrza dla zdrowia środowiska.
- Studenci zrozumieją, w jaki sposób zanieczyszczenia powietrza mogą wpływać na ekosystemy i zdrowie ludzkie.
- Uczniowie zbudują prosty czujnik jakości powietrza oparty na Arduino i będą zbierać dane.
- Studenci omówią znaczenie monitorowania jakości powietrza dla dobrostanu ekologicznego.

Materiały:

- Tablice Arduino Uno (po jednej na ucznia lub grupę)
- Moduł czujnika jakości powietrza (np. MQ-135)
- Przewody połączeniowe
- Kable USB do programowania
- Komputery z zainstalowanym Arduino IDE
- Projektor lub tablica do demonstracji

Zajęcia

Dzień 1

Wprowadzenie do jakości powietrza (10 minut):

- Rozpocznij lekcję od omówienia znaczenia czystego powietrza i jego wpływu na ekosystemy.
- Wyjaśnij, w jaki sposób zanieczyszczenia powietrza mogą wpływać zarówno na środowisko naturalne, jak i na zdrowie człowieka.

Czujniki jakości powietrza (20 minut):

- Przedstaw czujniki jakości powietrza i ich rolę w monitorowaniu zanieczyszczeń powietrza.
- Wyjaśnij podstawowe działanie czujników jakości powietrza i sposób, w jaki mierzą one różne gazy.

Podstawy Arduino (10 minut):

- Przedstaw Arduino jako platformę mikrokontrolera do gromadzenia danych.
- Pokaż uczniom elementy płytki Arduino oraz podstawy pisania i przesyłania kodu.

Przygotowanie do ćwiczeń praktycznych (15 minut):

- Zapewnij każdej grupie Arduino Uno, moduł czujnika jakości powietrza i przewody połączeniowe.
- Poleć uczniom, aby złożyli czujnik jakości powietrza i podłączyli go do Arduino.

Budowanie i programowanie czujnika jakości powietrza (20 minut):

- Poinstruuj uczniów, jak budować system czujników jakości powietrza oparty na Arduino.
- Pokaż uczniom, jak napisać kod Arduino w celu gromadzenia danych o jakości powietrza z czujnika.

Oto prosty przykład kodu Arduino do monitorowania jakości powietrza za pomocą czujnika jakości powietrza MQ-135. Ten kod odczytuje dane z czujnika i wyświetla je na monitorze szeregowym. Upewnij się, że masz zainstalowane odpowiednie biblioteki w Arduino IDE, ponieważ czujnik MQ-135 może wymagać kalibracji w celu uzyskania dokładnych wyników.

```
// Include necessary libraries
#include <Wire.h>
#include <Adafruit_Sensor.h>
#include <Adafruit_MQ135.h>

// Define the analog pin where the MQ-135 sensor is connected
#define MQ135_PIN A0

// Create an instance of the Adafruit MQ135 class
Adafruit_MQ135 mq135(MQ135_PIN);

void setup() {
 // Initialize serial communication for data output
 Serial.begin(9600);
}

void loop() {
 // Read the MQ-135 sensor's data
 float airQuality = mq135.readCO2();

 // Print the air quality data to the Serial Monitor
 Serial.print("Air Quality: ");
 Serial.print(airQuality);
 Serial.println(" ppm");

 // Add a delay before the next reading
 delay(2000); // Adjust the delay time as needed
}
```

W tym kodzie:
Dołączamy niezbędne biblioteki, w tym biblioteki Adafruit Sensor i Adafruit MQ135, które są powszechnie wykorzystywane do współpracy z czujnikiem jakości powietrza MQ-135. Upewnij się, że masz te biblioteki zainstalowane w Arduino IDE.
Definiujemy pin analogowy (w tym przykładzie A0), na którym podłączony jest czujnik MQ-135 do Arduino.
Tworzona jest instancja klasy Adafruit_MQ135 w celu interakcji z czujnikiem.
W funkcji setup() inicjujemy komunikację szeregową w celu wyprowadzenia danych do monitora szeregowego.
W funkcji pętli() na bieżąco odczytujemy dane o jakości powietrza z czujnika MQ-135 za pomocą mq135.readCO2(). Dane przedstawiają stężenie CO2 w częściach na milion (ppm).
Dane dotyczące jakości powietrza są drukowane na monitorze szeregowym, a przed wykonaniem kolejnego odczytu następuje 2-sekundowe opóźnienie (regulowane).
Należy pamiętać, że czujnik MQ-135 może wymagać kalibracji w celu uzyskania dokładnych odczytów, a podany tutaj kod służy jako podstawowy przykład. W zależności od konkretnego zastosowania i wymagań kalibracyjnych może być konieczne odpowiednie dostosowanie kodu i procesu kalibracji.

Gromadzenie danych i dyskusja (20 minut):

- Poinstruuj uczniów, aby zbierali dane dotyczące jakości powietrza, uruchamiając czujniki w różnych środowiskach (np. w pomieszczeniu, w pobliżu drogi, w ogrodzie).
- Poproś uczniów, aby zapisali i udostępnili swoje dane klasie.
- Poprowadź dyskusję w klasie na temat różnic w danych dotyczących jakości powietrza i potencjalnych konsekwencji ekologicznych.

Oceny

Oceniaj uczniów na podstawie ich udziału, gromadzenia danych i umiejętności omawiania wpływu jakości powietrza na systemy ekologiczne.

Praca domowa

Przypisz Praca domowa, która wymaga od studentów zbadania i przedstawienia rzeczywistych przykładów problemów ekologicznych związanych z zanieczyszczeniem powietrza oraz znaczenia monitorowania jakości powietrza w łagodzeniu tych problemów.

Wniosek

Zakończ lekcję, podsumowując kluczowe poznane koncepcje ekologiczne, podkreślając rolę technologii (Arduino) w monitorowaniu środowiska i zachęcając uczniów do rozważenia znaczenia jakości powietrza dla dobrostanu ekologicznego.

FROM GAME TO STEM

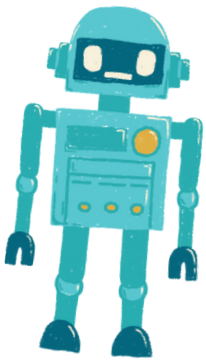
With the support of the Erasmus+ Programme of the European Union

Урок 1
Урок по математика с Ардуино

Тригонометрия с Ардуино

Продължителност:

3 периода



Цел:

- Учениците ще разберат основни тригонометрични концепции, включително синус, косинус и тангенс.
- Учениците ще прилагат тригонометрични функции за решаване на проблеми от реалния свят.
- Учениците ще програмират ардуино, за да създадат просто устройство за измерване на ъгли, използвайки серво мотор.

Материали:

- ардуино платки (по една на ученик или група)
- Серво мотори
- Експериментални пла (breadboards)
- Свързващи проводници (jum wires)
- Транспортори (протрактори)
- Линейки
- USB кабели за програмиране
- Компютри с инсталирана ардуино IDE
- Проектор или дъска за демонстрации

Активности

Ден 1

<u>Въведение в тригонометрията (15 минути):</u>

- Започнете урока, като представите концепцията на тригонометрията и нейното значение в реални приложения.
- Кратко обяснете синус, косинус и тангенс като отношения между страните в правоъгълен триъгълник.
- Обсъдете как тригонометричните функции могат да се използват за решаване на проблеми, свързани с ъгли и разстояния.

<u>Основи на тригонометрията (30 минути):</u>

- Погледнете по-дълбоко в дефинициите на синус, косинус и тангенс.
- Предоставете примери, как тези функции се използват за намиране на липсващи ъгли или дължини на страни в правоъгълни триъгълници.
- Разрешете на учениците да практикуват решаването на основни тригонометрични задачи на хартия.

<u>Въведение в ардуино и серво моторите (20 минути):</u>

- Представете ардуино като платформа за създаване на гаджети и обяснете нейните компоненти (микроконтролер, сензори, актуатори).
- Обяснете концепцията на серво моторите и как те могат да бъдат контролирани, за да се движат към конкретни ъгли.
- Покажете примери за управление на серво мотори с ардуино.

<u>Практическа дейност (45 минути):</u>

- Разделете учениците на двойки или малки групи.
- Предоставете на всяка група ардуино платка, серво мотор, експериментална платка и свързващи проводници.
- Наставете учениците да сглобят просто устройство за измерване на ъгли с помощта на серво мотор и транспортор като референция.
- Ръководете учениците в написването на ардуино код за управление на серво мотора, за да се движи към определен ъгъл в зависимост от вход от потребителя.

Ден 2

Преглед на концепциите на тригонометрията (20 минути):

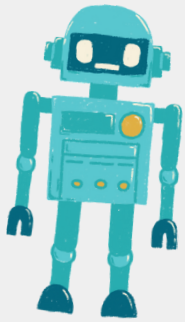
- Започнете втория ден, като прегледате концепциите за синус, косинус и тангенс, научени на първия ден.
- Предоставете допълнителни упражнения на учениците, които да решават, използвайки тригонометрични функции.

Програмиране на устройството за измерване на ъгли (45 минути):

- Продължете с практическата дейност от първия ден.
- Насочете учениците да завършат своя ардуино код, за да измерват и показват точни ъгли, използвайки серво мотор и числов дисплей (например, Serial Monitor).
- Поощрете учениците да прилагат тригонометрията, за да преобразуват позицията на серво мотора в ъгли.

Ето един примерен ардуино код за създаване на просто устройство за измерване на ъгли, използвайки серво мотор. Този код позволява на серво мотора да се движи към определен ъгъл в зависимост от вход от потребителя и да извежда измерения ъгъл на Сериски мотор.

```
#include <Servo.h>
Servo myservo; // Create a Servo object
void setup() {
myservo.attach(9); // Attaches the servo to digital pin 9
Serial.begin(9600); // Initialize serial communication for debugging
}
void loop() {
int angle; // Variable to store the desired angle
Serial.println("Enter an angle (0-180): ");
while (!Serial.available()) {
  // Wait for user input
}
angle = Serial.parseInt(); // Read the angle entered by the user
if (angle >= 0 && angle <= 180) {
// Check if the entered angle is within the valid range
myservo.write(angle); // Move the servo to the specified angle
delay(500); // Delay for stability
Serial.print("Measured angle: ");
Serial.print(angle);
Serial.println(" degrees");
} else {
Serial.println("Invalid angle. Please enter a value between 0 and 180.");
}
}
```



В този код:

1. **Включваме библиотеката Servo,** за да контролираме серво мотора.
2. В функцията **setup(),** прикачваме серво мотора към цифров пин 9 и инициализираме серийна комуникация за отстраняване на грешки (debugging).
3. В функцията **loop(),** четем входните данни на потребителя от Serial Monitor за желания ъгъл. Потребителят се подканва да въведе ъгъл между 0 и 180 градуса.
4. Проверяваме дали въведеният ъгъл се намира в валидния обхват (между 0 и 180 градуса). Ако е валиден, преместваме серво мотора на зададения ъгъл с командата myservo.write(angle) и извеждаме измерения ъгъл на Serial Monitor.
5. Ако въведеният ъгъл е извън валидния обхват, извеждаме съобщение за грешка.

За да използвате този код с учениците си, уверете се, че те свържат серво мотора към цифров пин 9 на Ардуино и отворят Serial Monitor, за да въвеждат ъгли и наблюдават измерените ъгли, когато серво моторът се движи. Този код предоставя прост и ефективен начин за демонстрация на приложението на тригонометрията в реални ситуации, използвайки Ардуино.

Ден 3

Презентация на проекта и дискусия (60 минути):

- Всяка група представя своето устройство за измерване на ъгли с Ардуино на предния клас.
- Поощрете учениците да обяснят как приложиха тригонометрични концепции в своите проекти.
- Обсъдете реалните приложения на устройствата за измерване на ъгли и тригонометрията.
- Организирайте класна дискусия относно предизвикателствата и решенията, с които студентите се сблъскват по време на проекта.

Оценяване

"Оценете студентите въз основа на тяхното участие, качество на кода, точност на измерването на ъгъла и тяхната способност да обяснят тригонометричните принципи зад устройството си."

Домашна работа

Задайте домашна работа, която изисква от учениците да изследват и представят реално приложение на тригонометрията в различни области, като инженерство, физика или архитектура.

Заключение

"Завършете урока, като обобщите основните тригонометрични концепции, научени и подчертаете тяхното практическо приложение в разработката на устройства с Ардуино. Подчертайте връзката между математиката и технологията."

FROM GAME TO STEM

With the support of the Erasmus+ Programme of the European Union

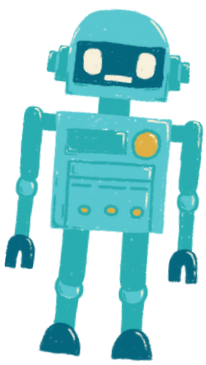
Урок 2

Ардуино Урок по Математика

Алгебрично моделиране с Ардуино

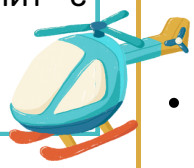
Продължителност:

2 учебни часа



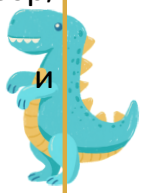
Цел:

- Учениците ще научат как да използват алгебрични концепции за решаване на задачи от реалния свят.
- Учениците ще приложат математическо моделиране и програмиране, за да създадат ардуино устройство, което решава практична задача.
- Учениците ще придобият опит с променливи и уравнения.



Материали:

- Ардуино платки (по един брой на ученик или група)
- Платки за експерименти (breadboards)
- Сензори или устройства за вход (например, температурен сензор, светлинен сензор, бутон)
- Светодиоди (LED), резистори и жички за връзка (jumper wires)
- USB кабели за програмиране
- Компютри с инсталирана среда за програмиране на Ардуино (Ардуино IDE)
- Проектор или бяла дъска за демонстрации



Активности

Ден 1

<u>Въведение в Алгебрично Моделиране (15 минути):</u>

- Започнете урока, като представите алгебричното моделиране и неговата важност при решаването на задачи от реалния свят.
- Обсъдете важността на променливите и уравненията в математичното моделиране.

<u>Променливи и Уравнения (30 минути):</u>

- Прегледайте концепцията на променливите и как те се използват за представяне на неизвестни стойности.
- Въведете линейни уравнения (например, y=mx+b) като начин за моделиране на взаимоотношенията между променливите.
- Предоставете примери на задачи от реалния свят, които могат да бъдат представени с помощта на уравнения.

<u>Основи на Ардуино (20 минути):</u>

- Представете Ардуино като платформа за създаване на гаджети и обяснете нейните компоненти.
- Покажете примери на прости проекти с Ардуино и как променливите могат да се използват в програмирането.

<u>Практическа Дейност (45 минути):</u>

- Разделете учениците на двойки или малки групи.
- Предоставете на всяка група Ардуино платка, breadboard, сензор или устройство за вход (например, температурен сензор) и LED.
- Инструктирайте учениците да създадат прост проект с Ардуино, който използва сензор за четене на данни и LED за визуално представяне на данните.
- Поощрете учениците да използват променливи за съхранение на данните от сензора и да създават уравнения, които да управляват LED в зависимост от данните от сензора.
- Обсъждайте различни сценарии от реалния свят, където техният гаджет може да бъде полезен.

Ден 2

Преглед на Алгебрични Концепции (20 минути):

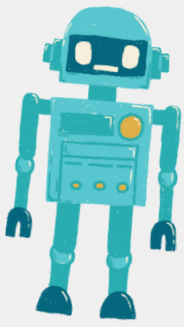
- Започнете втория ден с преглед на концепциите за променливи, уравнения и математично моделиране.
- Обсъдете проектите с Ардуино от предния ден и тяхното приложение.

Програмиране на Алгебричния Модел (45 минути):

- Продължете с практическата дейност от Първи Ден.
- Инструктирайте учениците да модифицират своя Ардуино код, за да създадат математичен модел, използвайки данните от сензора.
- Поощрете ги да експериментират с различни уравнения и променливи, за да представят взаимоотношението между данните от сензора и изхода на LED.
- Обсъдете как моделът може да бъде използван за прогнози или управление на системи в реалния свят.

Ето пример на Ардуино код за прост проект по алгебрично моделиране. В този проект учениците ще използват температурен сензор, за да четат температурни данни и да контролират LED в зависимост от температурното измерване. Кода използва променливи и уравнение, за да определи кога LED трябва да се включи или изключи.

```
// Define the pins for the temperature sensor, LED, and a resistor (if needed)
const int temperatureSensorPin = A0; // Analog pin for the temperature sensor
const int ledPin = 13; // Use the built-in LED on most ардуино boards
const float thresholdTemperature = 25.0; // Define the threshold temperature
void setup() {
pinMode(ledPin, OUTPUT); // Set the LED pin as an output
Serial.begin(9600); // Initialize serial communication for debugging (optional)
}
void loop() {
// Read the temperature from the sensor
int sensorValue = analogRead(temperatureSensorPin);
// Convert the analog value to temperature in degrees Celsius
float temperatureCelsius = map(sensorValue, 0, 1023, 0, 100); // Adjust the mapping as
needed
// Check if the temperature is above the threshold
if (temperatureCelsius > thresholdTemperature) {
digitalWrite(ledPin, HIGH); // Turn the LED on
} else {
digitalWrite(ledPin, LOW); // Turn the LED off
}
// Print the temperature to the serial monitor for debugging (optional)
Serial.print("Temperature: ");
Serial.print(temperatureCelsius);
Serial.println(" °C");
// Delay for a moment to avoid rapid LED toggling
delay(1000); // Delay for 1 second
}
```



В този код:
1. Дефинираме пиновете за температурния сензор (свързан към аналогов пин), светодиода (обикновено вграден LED на повечето Ардуино платки) и праговата температура, която определя кога светодиодът трябва да се включи.
2. В функцията setup() задаваме светодиодния пин като изход и инициализираме серийната комуникация за дебъг (опционално).
3. В функцията loop() четем температурата от сензора, използвайки analogRead(). Преобразуваме аналоговата стойност на сензора в температура в градуси Целзий, базирана на характеристиките на сензора.
4. След това проверяваме дали температурата надхвърля зададения праг (thresholdTemperature). Ако температурата е над прага, светодиодът се включва; в противен случай се изключва.
5. Опционално: Извеждаме температурата на Serial Monitor за цели на дебъг.
За да използвате този код, учениците трябва да свържат температурния сензор (например LM35) към аналогов пин на Ардуино и светодиода към цифров пин. Кодът чете температурата от сензора и управлява светодиода в зависимост от измерваната температура и стойността на прага.

Презентация на Проектите и Дискусия (30 минути):

- Всяка група представя своето Ардуино устройство на класа.
- Поощрете учениците да обяснят математичния модел, който създадоха, и как работи устройството.
- Улеснете класна дискусия относно приложенията на алгебричното моделиране в решаването на практични задачи.

Оценяване

Оценете учениците въз основа на техните участие, функционалността на техния Ардуино гаджет и техните умения да обяснят алгебричните концепции, използвани в техния проект.

Домашна работа

Задайте задача за Домашна работа, която изисква от учениците да изследват и представят реална проблем, който може да бъде решен с помощта на алгебрично моделиране и Ардуино устройство.

Заключение

Завършете урока, като обобщите основните алгебрични концепции, научени и подчертаете важността на математичното моделиране в технологията и инженерството.

FROM GAME TO STEM

Co-funded by the Erasmus+ Programme of the European Union

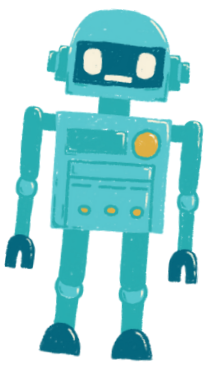
Урок 3

Урок по физика с Ардуино

Разглеждане на Движение и Ускорение с Ардуино

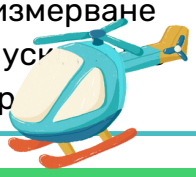
Продължителност:

3 периода



Обективен:

- Студентите ще разберат основните принципи на движението и ускорението.
- Студентите ще прилагат кинематични уравнения за решаване на проблеми от реалния свят, свързани с движението.
- Учениците ще програмират Ардуино за измерване и показване на уск с помощта на сензор



Материали:

- Ардуино дъски (по една на ученик или група)
- Сензорни модули за акселерометър (напр. ADXL345)
- Бредборди
- Джъмперни проводници
- USB кабели за програмиране
- Компютри с инсталиран Ардуино IDE
- Проектор или бяла дъска за демонстрации
- По избор: Обекти от реалния живот за експерименти с движение (напр. коли играчки, топки)

Занимания

Ден 1

<u>Въведение в движението и ускорението (15 минути):</u>

- Започнете урока, като представите концепцията за движение и ускорение.
- Дефинирайте ключови термини като скорост, ускорение и забавяне.
- Обсъдете значението на разбирането на движението в различни области, като физика, инженерство и транспорт.

<u>Физика на движението (30 минути):</u>

- Обяснете уравненията на движението, включително:
-
- Изместване: $s = ut + \frac{1}{2}at^2$
- Скорост: $v = u + at$
-
- Ускорение: $a = \frac{v-u}{t}$
- Дайте прим ова как тези уравнения се използват за анализ на движението.
- Провеждайте прости експерименти с движение (напр. търкаляне на топка надолу по склон), за да демонстрирате принципите на движение.

<u>Въведение в Ардуино и акселерометрите (20 минути):</u>

- Представете Ардуино като платформа за създаване на джаджи и обяснете неговите компоненти.
- Обяснете концепцията за акселерометрите и как те измерват ускорението.
- Покажете примери за извеждане на данни от акселерометъра и обяснете каква е връзката им с движението в реалния свят.

<u>Практическа дейност (45 минути):</u>

- Разделете учениците на двойки или малки групи.
- Осигурете на всяка група платка Ардуино, сензорен модул за акселерометър, макетна платка и джъмпери.
- Инструктирайте учениците да сглобят проста верига за свързване на акселерометъра към Ардуино.
- Насочвайте учениците да пишат код на Ардуино, за да четат и показват данни за ускорение от сензора в серийния монитор.

Ден 2

Преглед на кинематичните уравнения (20 минути):

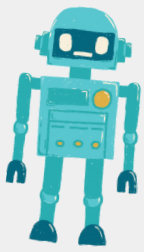
- Започнете втория Ден, като прегледате кинематичните уравнения, обсъдени в Ден 1.
- Осигурете допълнителни примери за решаване на учениците с помощта на тези уравнения.

Програмиране на устройството за измерване на ускорението (45 минути):

- Продължете с практическата дейност от Ден 1.
- Инструктирайте учениците да попълнят своя код на Ардуино, за да четат и показват данни за ускорение в реално време от акселерометъра.
- Насърчете учениците да калибрират сензора, ако е необходимо, и да приложат уравненията за ускорение, за да интерпретират данните.

Ето пример за код на Ардуино за измерване и показване на ускорение с помощта на сензор за акселерометър ADXL345. Този код ще позволи на вашите ученици да свързват сензора на акселерометъра с Ардуино и да показват стойностите на ускорението на серийния монитор.

```
#include <Wire.h>
#include <Adafruit_Sensor.h>
#include <Adafruit_ADXL345_U.h>
Adafruit_ADXL345_Unified accel = Adafruit_ADXL345_Unified(12345);
void setup(void) {
 Serial.begin(9600);
 Serial.println("Accelerometer Test"); Serial.println("");
  if(!accel.begin())
 {
/* There was a problem detecting the ADXL345 ... check your connections*/
   Serial.println("Ooops, no ADXL345 detected ... Check your wiring!");
   while(1);
 }
}
void loop(void) {
 sensors_event_t event;
 accel.getEvent(&event);
 /* Display the acceleration values (in m/s^2) */
 Serial.print("X: "); Serial.print(event.acceleration.x); Serial.print(" ");
 Serial.print("Y: "); Serial.print(event.acceleration.y); Serial.print(" ");
 Serial.print("Z: "); Serial.print(event.acceleration.z); Serial.println(" m/s^2");
 delay(500); // Delay for half a second between readings
}
```



В този код:
Включваме необходимите библиотеки за сензора за акселерометър ADXL345.
Ние създаваме Adafruit_ADXL345_Unified обект, наречен accel, за да взаимодейства със сензора.
Във функцията setup() инициализираме серийната комуникация за отстраняване на грешки и проверяваме дали сензорът на акселерометъра е открит. Ако не, ще покаже съобщение за грешка.
Във функцията loop() ние непрекъснато четем данните за ускорението от сензора с помощта на accel.getEvent(&event) и показваме стойностите на ускорението по оста X, Y и Z в метри в секунда на квадрат (m/s^2) на Сериен монитор.
За да използвате този код, уверете се, че вашите ученици са свързали правилно сензора за акселерометър ADXL345 към Ардуино. Сензорът трябва да бъде свързан към подходящите щифтове (напр. SDA и SCL) за I2C комуникация. Когато Ардуино е включен и изпълнява този код, той непрекъснато ще показва данните за ускорението на серийния монитор.
Моля, обърнете внимание, че може да се наложи да инсталирате библиотеката Adafruit ADXL345 чрез Ардуино Library Manager, за да използвате този код.

Представяне на проекта и дискусия (60 минути):

- Всяка група представя своето базирано на Ардуино устройство за измерване на ускорението пред класа.
- Насърчете учениците да обяснят как са приложили кинематичните уравнения и физичните принципи в своите проекти.
- Обсъдете приложенията в реалния свят на устройствата за измерване на ускорението в различни области.
- Улеснете дискусия в клас относно предизвикателствата и решенията, възникнали по време на проекта.

Оценки

Оценявайте учениците въз основа на тяхното участие, качество на кода, точност на измерване на ускорението и способността им да обяснят физичните принципи зад тяхната притурка.

Домашна работа

Задайте задача за Домашна работа, която изисква от учениците да изследват и представят реално приложение на измерване на ускорението във физиката или инженерството

Заключение

Завършете урока, като обобщите научените ключови физични концепции и наблегнете на практическото им използване при разработването на джаджи Ардуино. Подчертайте връзката между физиката и технологиите.

FROM GAME TO STEM

Co-funded by the Erasmus+ Programme of the European Union

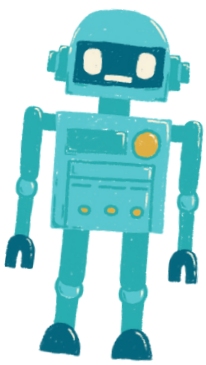
Урок 4

Урок по физика на Ардуино

Изследване на трептения и движение на махало с Ардуино

Продължителност:

2 периода



Обективен:

- Студентите ще разберат принципите на колебателното движение и хармоничните трептения.
- Студентите ще прилагат математическо моделиране, за да анализират движението на махалото.
- Студентите ще програмират Ардуино за симулиране и визуализиране на движението на махалото.

Материали:

- Ардуино дъски (по една на ученик или група)
- Серво мотори
- Бредборди
- Джъмперни проводници
- Малки предмети като тежести на махалото (напр. шайби)
- Връв или конец за махала
- Линийки или измервателна лента
- USB кабели за програмиране
- Компютри с инсталиран Ардуино IDE
- Проектор или бяла дъска за демонстрации

Занимания

Ден 1

<u>Въведение в трептенията и движението на махалото (15 минути):</u>

- Започнете урока, като представите концепцията за осцилаторно движение и нейното значение в различни области, включително физика и инженерство.
- Дефинирайте ключови термини като период, честота, амплитуда и хармонични трептения.
- Обяснете основите на движението на махалото и неговото математическо представяне.

<u>Математическо моделиране на махала (30 минути):</u>

- Обсъдете математическия модел на просто махало, включително уравнението за периода на махалото:
- където T е периодът, L е $T = 2\pi\sqrt{\frac{L}{g}}$ махалото и g е ускорението, дължащо се на гравитацията.
- Дайте примери как да използвате уравнението за изчисляване на периода на махало.
- Извършвайте прости изчисления, свързани с движението на махалото.

<u>Въведение в Ардуино и серво моторите (20 минути):</u>

- Представете Ардуино като платформа за създаване на джаджи и обяснете неговите компоненти (микроконтролер, сензори, изпълнителни механизми).
- Обяснете концепцията за серво мотори и как те могат да бъдат използвани за симулиране на движение на махало.
- Покажете примери за управление на серво мотор с Ардуино.

<u>Практическа дейност (45 минути):</u>

- Разделете учениците на двойки или малки групи.
- Осигурете на всяка група платка Ардуино, серво мотор, макетна платка, джъмпери, малък предмет като тежест на махало и връв.
- Инструктирайте учениците да сглобят прост симулатор на махало, като използват серво мотора за управление на движението на махалото.
- Насочвайте учениците да пишат код на Ардуино за създаване на хармонични трептения чрез промяна на позицията на сервото.

Ден 2

Преглед на движението на махалото (20 минути):

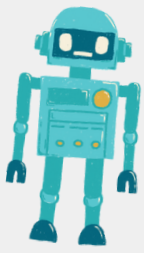
- Започнете втория Ден, като прегледате концепциите за колебателно движени движение на махало и математическия модел на просто махало.
- Обсъдете проектите на Ардуино от предишния ден и техните приложения.

Програмиране на симулатора на махало (45 минути):

- Продължете с практическата дейност от Ден 1.
- Инструктирайте учениците да променят своя код на Ардуино, за да симулират точно движение на махало с различни параметри като дължина и амплитуда.
- Насърчете учениците да визуализират и представят графично движението с помощта на серво мотора.

Ето пример за код на Ардуино за създаване на прост симулатор на махало с помощта на серво мотор. Този код позволява на учениците да програмират Ардуино, за да симулират и визуализират движението на махало:

```
#include <Servo.h>
Servo pendulum; // Create a Servo object
int angle = 90; // Initial angle of the pendulum (straight down)
int amplitude = 45; // Maximum angle to one side of the vertical (adjust as needed)
int period = 2000; // Period of the pendulum swing in milliseconds (adjust as needed)
void setup() {
  pendulum.attach(9); // Attach the servo to digital pin 9
}
void loop() {
  // Calculate the angle of the pendulum using harmonic motion
  int displacement = amplitude * cos(2 * PI * millis() / period);
  int pendulumAngle = angle + displacement;
   // Move the servo to the calculated angle
  pendulum.write(pendulumAngle);
  // Delay for a short time to control the speed of the simulation
  delay(50); // Adjust as needed for desired speed
}
```



В този код:
Включваме Servo библиотеката за управление на серво мотора.
Създаваме серво обект, наречен махало, за да контролираме серво мотора.
Ние дефинираме променливи за началния ъгъл на махалото (ъгъл), максималния ъгъл към едната страна на вертикалата (амплитуда) и периода на люлеене на махалото (период). Можете да коригирате тези стойности, за да промените поведението на махалото.
Във функцията setup() свързваме сервото към цифров щифт 9.
Във функцията loop() изчисляваме ъгъла на махалото, използвайки формулата за хармонично движение. Изместването представлява ъгловото изместване от вертикалната позиция и ние го добавяме към първоначалния ъгъл, за да получим ъгъла на махалото.
Използваме pendulum.write(pendulumAngle), за да преместим сервото до изчисления ъгъл.
Добавяме забавяне, за да контролираме скоростта на симулацията. Регулирайте стойността на забавянето според нуждите, за да постигнете желаната скорост на симулация.
За да използват този код, учениците трябва да свържат серво мотор към цифров пин 9 на Ардуино и да качат кода на платката. Серво моторът ще симулира движението на махало и учениците могат да наблюдават трептенията в действие.

Представяне на проекта и дискусия (30 минути):

- Всяка група представя своя базиран на Ардуино симулатор на махало пред класа.
- Насърчете учениците да обяснят как са приложили принципите на математическото моделиране в своите проекти.
- Обсъдете приложенията в реалния свят за разбиране на хармоничните трептения в области като физика, инженерство и астрономия.
- Улеснете дискусия в клас относно предизвикателствата и решенията, възникнали по време на проекта.

Оценки

Оценявайте учениците въз основа на тяхното участие, качеството на тяхната притурка Ардуино и способността им да обяснят принципите на осцилаторното движение и хармоничните трептения.

Домашна работа

Задайте задача за Домашна работа, която изисква от учениците да изследват и представят приложение в реалния свят на трептения и хармонично движение в науката или технологиите.

Заключение

Завършете урока, като обобщите научените ключови физични концепции и наблегнете на практическото им използване при разработването на джаджи Ардуино. Подчертайте връзката между математиката и технологиите в разбирането на движението на махалото.

FROM GAME TO STEM

Co-funded by the Erasmus+ Programme of the European Union

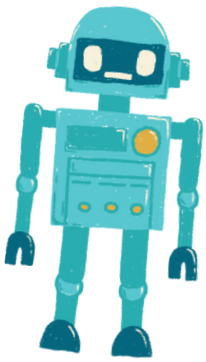
Урок 5

Урок по Ардуино за химията

Изследване на хидропониката и химията на растежа на растенията

Продължителност:

3 периода



Цел:

- Учениците ще разберат принципите на хидропониката и нейните предимства за растежа на растения.
- Учениците ще научат за основните хранителни вещества за растежа на растенията и как да подготвят хранителни разтвори.
- Учениците ще проектират и сглобят автоматизирана система за напояване с използване на Ардуино.
- Учениците ще следят и контролират нивата на pH в хидропонична систе[illegible] да оптимизират рас[illegible] на растенията.

Материали:

- Ардуино платки (по една на ученик или група)
- pH сензори и разтвори за калибриране на pH
- Водни помпи
- Шлаухи и капкови емитери
- Резервоар за хранителен разтвор
- Разтвори за повишаване и намаляване на pH
- Буферни разтвори за pH
- Семена или млади растения
- Хидропонична среда за отглеждане (например, каменна вата, перлит)
- Съставки за хранителния разтвор (например, N-P-K тор)
- Контейнери за хидропонични системи
- Свързващи проводници (jumper wires)
- USB кабели за програмиране
- Компютри с инсталирана Ардуино IDE
- Проектор или маркерна дъска за демонстрации

Дейности:

Ден:1

<u>Въведение в Хидропониката (15 минути):</u>

- Започнете урока, като представите концепцията на хидропониката и нейните предимства за растежа на растенията.
- Обсъдете ползите от контролирани среди и водоспестяващи системи.

<u>Химия на Растежа на Растенията (30 минути):</u>

- Обяснете химичните елементи, необходими за растежа на растенията (например, азот, фосфор, калий) и техните функции.
- Обсъдете важността на нивата на pH за усвояването на хранителните вещества и здравето на растенията.
- Въведете концепцията за хранителни разтвори и контрол на pH в хидропониката.

<u>Въведение в Ардуино и Сензори (20 минути):</u>

- Представете Ардуино като платформа за автоматизация и събиране на данни и обяснете нейните компоненти (микроконтролер, сензори).
- Покажете примери на сензори, използвани в хидропонични системи (например, pH сензори).

Подготовка за Практическа Дейност (45 минути):

- Дайте на учениците Ардуино платки, pH сензори, водни помпи, шлаухи и контейнери.
- Научете учениците да направят мозайка и да планират дизайна на своята хидропонична система, включително хранителни разтвори и контрол на pH.

Ден:2

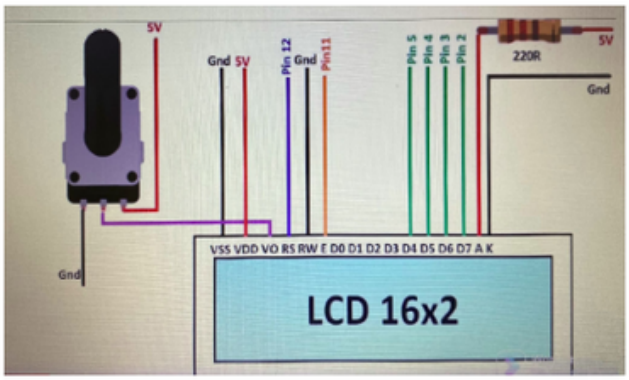
Настройка на Хидропоничната Система (60 минути):

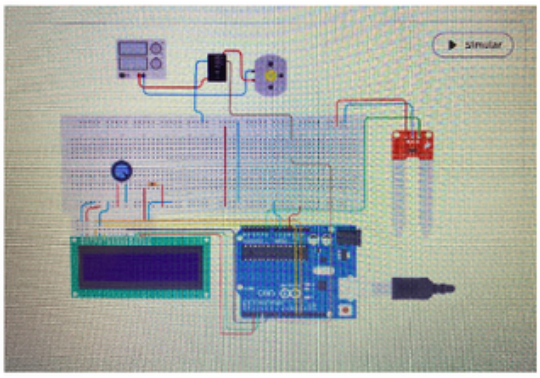
- Започнете втория ден, като позволите на учениците да настроят своите хидропонични системи.
- Учениците трябва да засадят семена или млади растения в хидропонична среда за отглеждане (например, каменна вата) и да настроят системата за напояване с помощта на шлаухи и капкови емитери.
- Демонстрирайте как се смесва и приготвя хранителни разтвори.

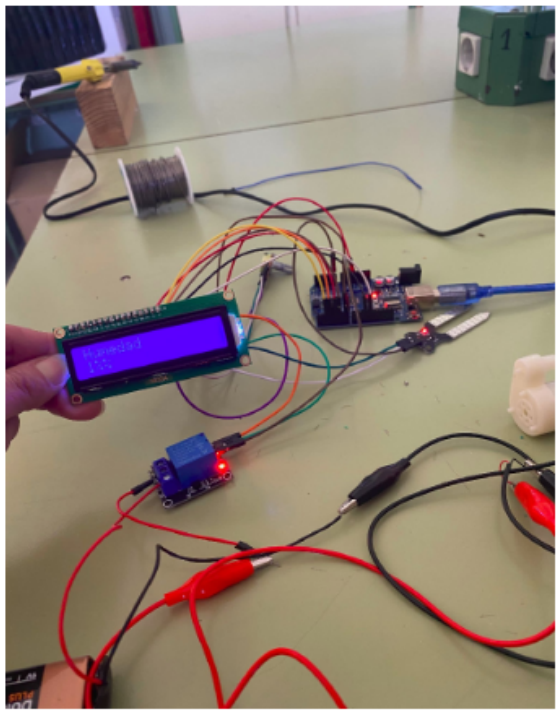
Мониторинг и Калибриране на pH (30 минути):

- Инструктирайте учениците как да калибрират и използват pH сензорите за мониторинг на pH.
- Обяснете важността на поддържането на pH в оптималния диапазон за усвояване на хранителни вещества (обикновено около pH 6-7).
- Ръководете учениците при калибрирането на техните pH сензори, използвайки pH буферни разтвори.

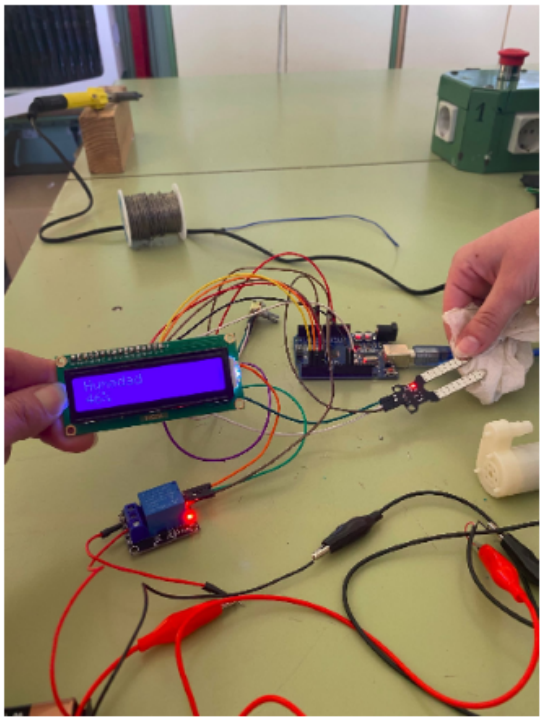
Схема на връзките:







1% влажност: помпата работи



46% влажност: помпата продължава да работи

Ден:3

Настройка на Хидропоничната Система (60 минути):

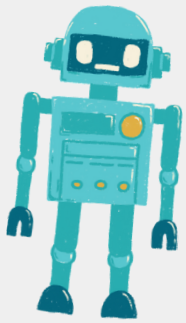
- Започнете втория ден, като позволите на учениците да настроят своите хидропонични системи.
- Учениците трябва да засадят семена или млади растения в хидропонична среда за отглеждане (например, каменна вата) и да настроят системата за напояване с помощта на шлаухи и капкови емитери.
- Демонстрирайте как се смесва и приготвя хранителни разтвори.

Мониторинг и Калибриране на pH (30 минути):

- Инструктирайте учениците как да калибрират и използват pH сензорите за мониторинг на pH.
- Обяснете важността на поддържането на pH в оптималния диапазон за усвояване на хранителни вещества (обикновено около pH 6-7).
- Ръководете учениците при калибрирането на техните pH сензори, използвайки pH буферни разтвори.

Ето един пример на Ардуино код за автоматизирана хидропонична система за напояване с мониторинг и контрол на pH. Този код е основна структура, която можете да адаптирате и разширите в зависимост от конкретната си настройка и нужди. Той включва мониторинг на pH, корекция на pH и контрол на водната помпа за напояване.

```
#include <Adafruit_ADS1015.h>
#include <Wire.h>
#define PH_SENSOR_PIN 0 // Analog pin for pH sensor
#define PUMP_PIN 12    // Digital pin for water pump
// Create an Adafruit ADS1015 ADC object
Adafruit_ADS1015 ads;
float currentpH;
float targetpH = 6.5; // Adjust this value to your desired pH level
void setup() {
 Serial.begin(9600);
 ads.begin();
 pinMode(PUMP_PIN, OUTPUT);
 digitalWrite(PUMP_PIN, LOW); // Initialize pump as off
}
void loop() {
 // Read pH value from pH sensor
 currentpH = readpH();
 // Display current pH value
 Serial.print("Current pH: ");
 Serial.println(currentpH);
 // Check and adjust pH level
 if (currentpH < targetpH) {
  // pH is too low, add pH up solution (adjust as needed)
  // Implement pH adjustment mechanism here
    // For example, you can control a peristaltic pump for adding pH up solution
```



```
digitalWrite(PUMP_PIN, HIGH); // Turn on the pump
 delay(1000); // Adjust the delay time as needed
 digitalWrite(PUMP_PIN, LOW); // Turn off the pump
 } else if (currentpH > targetpH) {
 // pH is too high, add pH down solution (adjust as needed)
 // Implement pH adjustment mechanism here
  // For example, you can control a peristaltic pump for adding pH down solution
 digitalWrite(PUMP_PIN, HIGH); // Turn on the pump
 delay(1000); // Adjust the delay time as needed
 digitalWrite(PUMP_PIN, LOW); // Turn off the pump
 }
 // Add code for irrigation control here (e.g., based on time intervals)
}
float readpH() {
 // Read pH value from the pH sensor and convert to pH scale
 int16_t rawValue = ads.readADC_SingleEnded(PH_SENSOR_PIN);
 float voltage = (rawValue * 0.1875) / 1000.0; // Convert to voltage
  float pHValue = 3.5 * voltage + 3.5; // Convert to pH scale (adjust calibration values as needed)
 return pHValue;
}
```

В този код:

Ние използваме библиотеката Adafruit ADS1015 за взаимодействие с ADS1015 ADC, който се използва за четене на стойности на pH сензора. Уверете се, че имате инсталирана тази библиотека във вашата Ардуино IDE.

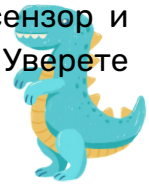
Функцията setup() инициализира серийна комуникация за извеждане на данни, инициализира ADC и конфигурира щифта на водната помпа като изход.

Във функцията loop() ние непрекъснато четем стойността на pH от сензора за pH, като използваме функцията readpH().

Ние сравняваме текущата стойност на pH с целевото ниво на pH (targetpH) и коригираме pH, ако е необходимо. Трябва да приложите механизма за регулиране на pH въз основа на вашата конкретна настройка, която може да включва контролиране на перисталтична помпа за добавяне на разтвори за повишаване или понижаване на pH.

Освен това можете да добавите код за управление на водната помпа за напояване въз основа на интервали от време или други критерии.

Моля, обърнете внимание, че този код предоставя основна рамка и може да се наложи да го калибрирате и фино настроите според вашия специфичен сензор и настройка на помпата, както и желаните нива на pH и графика за напояване. Уверете се, че са взети предпазни мерки при работа с химикали и помпи.



Ден:4

Експериментална Настройка и Мониторинг (60 минути):

- Позволете на учениците да настроят своите автоматизирани хидропонични системи в парницата или класната стая.
- Учениците трябва да стартират системата и да следят растежа на растенията, нивата на хранителните вещества и pH.

Представяне на Проекта и Дискусия (60 минути):

- Всяка група представя дизайна на своята хидропонична система, методологията и първите резултати пред класа.
- Обсъждайте влиянието на хранителните разтвори и контрола на pH върху растежа на растенията.
- Поощрете учениците да предложат оптимизации за своите системи, базирани на първите наблюдения.

Assessments

Оценете учениците на база техните участие, настройка на системата, калибриране на pH, събиране на данни, анализ и качество на техните презентации.

Homework

Задайте домашно на учениците, което изисква да изследват и представят реален приложение на хидропониката в селското стопанство или устойчивото земеделие.

Conclusion

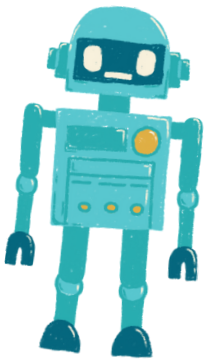
Завършете урока, като направите обобщение на ключовите химични концепции, които учениците научиха, и подчертайте значимостта на хидропониката в устойчивото селско стопанство и производство на храна. Акцентирайте ролята на технологията (Ардуино) за автоматизиране и оптимизация на хидропоничните системи.

Урок 6
Урок по Химия с Ардуино

Урок 6
Урок по Химия с Ардуино

Продължителност:

3 периода



Цел:

- Студентите ще разберат принципите на химичните реакции и тяхното значение в ежедневието.
- Студентите ще проектират и провеждат експерименти за наблюдение и измерване на химични реакции.
- Студентите ще програмират базирана на Ардуино система за наблюдение на температурата, за да събират и анализират данни от химични реакции.

Материали:

- Ардуино дъски (по една на ученик или група)
- Температурни сензори (напр. DS18B20 или LM35)
- Химикали за експерименти (напр. сода за хляб, оцет)
- Реакционни контейнери (напр. чаши, епруветки)
- Джъмперни проводници
- USB кабели за програмиране
- Компютри с инсталиран Ардуино IDE
- Проектор или бяла дъска за демонстрации
- Предпазни очила и лабораторни

Дейности:

Ден:1

<u>Въведение в химичните реакции (15 минути):</u>

- Започнете урока, като представите концепцията за химичните реакции и тяхната роля във всеки ден.
- Обсъдете значението на химичните реакции в различни области, включително химия, биология и индустрия.

<u>Основи на химическата реакция (30 минути):</u>

- Потопете се в основите на химичните реакции, включително реагентите, продуктите и запазването на масата.
- Дайте примери за общи химични реакции и техните приложения.
- Наблегнете на протоколите за безопасност при работа с химикали в експерименти.

<u>Въведение в Ардуино и сензорите (20 минути):</u>

- Представете Ардуино като платформа за събиране на данни и обяснете неговите компоненти (микроконтролер, сензори).
- Покажете примери за температурни сензори и как те могат да бъдат свързани към Ардуино за събиране на данни.

<u>Подготовка за практическа дейност (45 минути):</u>

- Осигурете на учениците платки Ардуино, температурни сензори, химикали и реакционни контейнери.
- Инструктирайте учениците да обмислят и планират своите експерименти с химични реакции, като се фокусират върху температурните промени.
- Обсъдете мерките за безопасност и лабораторните правила.

Ден:2

Настройка на експеримента и събиране на данни (60 минути):

- Започнете втория Ден: като позволите на учениците да организират своит експерименти с химически реакции.
- Учениците трябва да смесват избраните химикали в реакционни контейнери и да поставят температурните сензори.
- Инструктирайте учениците да напишат код на Ардуино за наблюдение на температурата и събиране на данни.

Анализ на данни и дискусия (30 минути):

- Насочвайте учениците да анализират своите данни, търсейки температурни промени по време на химичните реакции.
- Обсъдете значението на температурните промени в химичните реакции и понятията за екзотермични и ендотермични реакции.
- Насърчете учениците да правят заключения въз основа на своите открития.

Ден:3

Програмиране на системата за наблюдение (60 минути):

- Инструктирайте учениците да прецизират своя код на Ардуино за събиране и анализ на данни.
- Учениците трябва да програмират Ардуино да записва температурни данни на определени интервали и да ги показва графично (напр. на LCD екран или сериен монитор).

Представяне на проекта и дискусия (60 минути):

- Всяка група представя своята настройка, методология и резултати от експеримента с химична реакция пред класа.
- Насърчете учениците да обяснят своите наблюдения и последиците от температурните промени в химичните реакции.
- Улеснете дискусия в клас относно приложенията на химичните реакции в реалния свят в различни области.

Ето пример за код на Ардуино, който можете да използвате за наблюдение на температурата по време на експеримент с химическа реакция. Този код чете данни за температурата от температурен сензор DS18B20 и ги показва на серийния монитор. Можете да персонализирате и разширите този код, за да отговаря на конкретния ви експеримент.

```
#include <OneWire.h>
#include <DallasTemperature.h>
// Data wire is plugged into digital pin 2
#define ONE_WIRE_BUS 2
// Create a OneWire object
OneWire oneWire(ONE_WIRE_BUS);
// Pass the OneWire reference to Dallas Temperature sensor
DallasTemperature sensors(&oneWire);
```

```
void setup() {
 // Initialize serial communication for data output
 Serial.begin(9600);

 // Start the temperature sensor library
 sensors.begin();
}

void loop() {
 // Request temperature readings
 sensors.requestTemperatures();

 // Read temperature in Celsius
 float temperatureC = sensors.getTempCByIndex(0);

 // Display temperature on the Serial Monitor
 Serial.print("Temperature: ");
 Serial.print(temperatureC);
 Serial.println(" °C");

 // Add code here for data storage or further analysis

 // Delay before the next temperature reading (adjust as needed)
 delay(1000); // 1-second delay
}
```

В този код:

Ние използваме температурната библиотека на Dallas за взаимодействие с температурния сензор DS18B20. Уверете се, че имате инсталирана тази библиотека във вашата Ардуино IDE.
Функцията setup() инициализира серийна комуникация за извеждане на данни и стартира библиотеката на температурния сензор.
Във функцията loop() програмата непрекъснато изисква и чете данни за температурата от сензора с помощта на sensors.getTempCByIndex(0). 0 показва първия (и в този случай единствен) свързан температурен сензор.
Данните за температурата се показват на серийния монитор със закъснение от 1 секунда между отчитанията. Можете да регулирате времето на забавяне, за да контролирате честотата на събиране на данни.
Можете да модифицирате този код, за да включите допълнителни сензори за различни експерименти, да внедрите съхранение на данни на SD карта или да създадете графично представяне на данните на LCD екран, в зависимост от вашите специфични изисквания.

Оценки

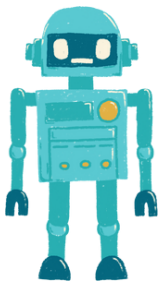
Оценявайте учениците въз основа на участието им в експеримента, събирането на данни, анализа и качеството на техните презентации.

Домашна работа

Задайте домашна работа, която изисква от учениците да изследват и представят реално приложение на химични реакции в конкретна индустрия или област на науката.

Заключение

Завършете урока, като обобщите научените ключови концепции по химия и затвърдите значението на химичните реакции за разбирането на природните процеси и технологичния напредък. Подчертайте ролята на технологията (Ардуино) в научните изследвания.



FROM GAME TO STEM

Co-funded by the Erasmus+ Programme of the European Union

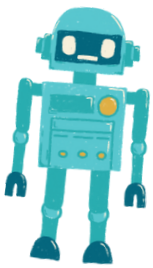
Урок 7

Урок по биология с Ардуино

Изследване на топлопроводимостта с термометри Ардуино

Продължителност:

3 периода



Обективен:

- Студентите ще разберат концепцията за топлопроводимост и нейното значение.
  Студентите ще проектират и проведат експеримент за определяне на топлопроводимостта на различни Материали:.
- Студентите ще използват термометри Ардуино , за да събират данни за температурата и да анализират своите открития.

Материали:

- Ардуино дъски (по една на ученик или група)
- Температурни сензори (напр. DS18B20 или LM35)
- Бредборди
- Джъмперни проводници
- Различни материали: за тестване на проводимост (напр. метали, пластмаси, дърво, стъкло)
- Изолационни материали: (напр. пяна, плат)
- Топла вода или източник на топлина
- Студена вода или лед
- Хронометър или таймер
- USB кабели за програмиране
- Компютри с инсталиран Ардуино IDE
- Проектор или бяла дъска за демонстрации
- Предпазни очила и ръкавици (за работа с горещи материали:)

Занимания

Ден 1

<u>Въведение в Топлопроводимостта (15 минути):</u>

- Започнете урока с представянето на концепцията за топлопроводимост и защо тя е важна в ежедневието.
- Обсъдете реални приложения на топлопроводимостта, като готвене, строителство и инженерия.

<u>Пренос на Топлина и Изолация (30 минути):</u>

- Обяснете различните методи на пренос на топлина (проводимост, конвекция, излъчване) и се фокусирайте върху проводимостта, която е важна за експеримента.
- Обсъдете изолационните материали и тяхната роля при намаляване на топлопреноса.
- Подчертайте необходимостта от контролиран експеримент за измерване на топлопроводимостта.

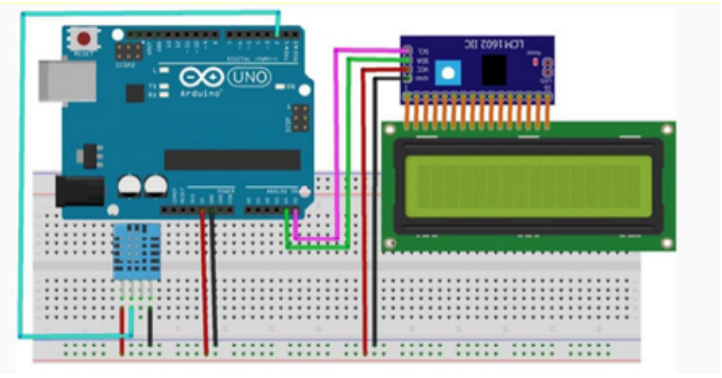
<u>Въведение в Ардуино Термометрите (20 минути):</u>

- Представете Ардуино като платформа за събиране на данни и обяснете нейните компоненти (микроконтролер, сензори).
- Обяснете използването на температурни сензори и как те могат да бъдат свързани към Ардуино платки.
- Покажете примери за събиране на данни за температура и визуализация с Ардуино .

<u>Подготовка за Практическа Дейност (45 минути):</u>

- Предоставете на учениците Ардуино платки, температурни сензори, експериментални платки и жички за свързване.
- Обяснете експерименталния сетъп: Всяка група ще изпитва различни материали, за да определи тяхната топлопроводимост.
- Указания на учениците да измислят и планират своите експерименти, включително начина на контролиране на променливите и събиране на данни.

ШЕМА:



Ден 2

Настройка на Експеримента и Събиране на Данни (60 минути):

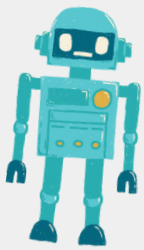
- Започнете втория ден, като позволите на учениците да настроят своите експерименти.
- Учениците трябва да поставят температурния сензор върху избрания от тях материал и да подготвят контейнери с гореща и студена вода.
- Инструктирайте учениците да започнат събирането на данни за температурата с помощта на Ардуино и да записват началните температури.
- Поискайте от учениците да измерят и записват времето, за което температурата се променя във всеки материал.

Анализ на Данните и Обсъждане (30 минути):

- Насочете учениците как да анализират своите данни, как да изчислят скоростта на промяна на температурата и как да сравнят проводимостта на различни материали.
- Обсъдете концепцията за топлопроводимост и как тя се свързва с термалната резистентност.

Ето пример за код на Ардуино , който можете да използвате за експеримента за измерване и сравняване на топлопроводимостта на различни материали: използване на температурен сензор (напр. DS18B20). Този код ще събира данни за температурата от сензора и ще ви позволи да изчислите скоростта на промяна на температурата за всеки материал.

```
#include <OneWire.h>
#include <DallasTemperature.h>
// Data wire is plugged into digital pin 2
#define ONE_WIRE_BUS 2
// Create a OneWire object
OneWire oneWire(ONE_WIRE_BUS);
// Pass the OneWire reference to Dallas Temperature sensor
DallasTemperature sensors(&oneWire);
void setup() {
 // Initialize serial communication for data output
 Serial.begin(9600);
 // Start the temperature sensor library
 sensors.begin();
}
void loop() {
```



```
// Initialize variables for temperature measurements
 float initialTemp, finalTemp;
 unsigned long startTime, endTime;
 float rateOfChange;
 // Wait for the user to start the experiment (e.g., press a button)
while (!digitalRead(3)) {
  // Wait for button press
 }
 delay(1000); // Debounce delay
 // Measure initial temperature
 sensors.requestTemperatures(); // Request temperature readings
 initialTemp = sensors.getTempCByIndex(0); // Get temperature in Celsius
 // Record the start time
 startTime = millis();
 // Wait for the temperature to stabilize (e.g., 5 seconds)
 delay(5000);
 // Measure final temperature
 sensors.requestTemperatures();
 finalTemp = sensors.getTempCByIndex(0);
 // Record the end time
 endTime = millis();
 // Calculate the rate of temperature change (°C per second)
  rateOfChange = (finalTemp - initialTemp) / ((endTime - startTime) /
1000.0);
 // Output results to the Serial Monitor
 Serial.print("Initial Temperature: ");
 Serial.print(initialTemp);
 Serial.println(" °C");
 Serial.print("Final Temperature: ");
 Serial.print(finalTemp);
 Serial.println(" °C");
 Serial.print("Rate of Change: ");
 Serial.print(rateOfChange);
 Serial.println(" °C/s");
 // Wait for user input (e.g., press a button) to proceed to the next material
 Serial.println("Press a button to test the next material.");
 while (!digitalRead(3)) {
  // Wait for button press
 }
 delay(1000); // Debounce delay
}
```

В този код:

1. Използваме библиотеката Dallas Temperature, за да се свържем с температурния сензор DS18B20. Уверете се, че имате тази библиотека инсталирана във вашия Ардуино IDE.
2. Задаваме цифровия пин (например, пин 2), към който е свързан данъчният кабел на сензора DS18B20.
3. В функцията setup() инициализираме сериалната комуникация за изход на данни и стартираме библиотеката за температурния сензор.
4. В функцията loop() програмата изчаква натискането на бутон, за да стартира експеримента. Температурата се измерва преди и след изчаквателния период (например, 5 секунди), за да се изчисли скоростта на промяната на температурата.
5. Скоростта на промяната се изчислява като разликата в температурата, разделена на времето, необходимо за постигане на тази промяна. Това ви дава скорост в градуси Целзий на секунда.
6. Резултатите се отпечатват в Serial Monitor, включително началната температура, крайната температура и скоростта на промяната.
7. След като експериментът за даден материал приключи, можете да натиснете бутон, за да продължите с следващия материал.

Ден 3

Представяне на проекта и дискусия (60 минути):

- Всяка група представя своята експериментална настройка, методология и резултати пред класа.
- Насърчете учениците да обяснят своите наблюдения, да обсъдят потенциални източници на грешки и да предложат подобрения.
- Улеснете дискусия в клас относно реалните последици от техните открития, като например избор на Материали: за топлоизолация или провеждане на Материали: за радиатори.

Оценки

Оценявайте учениците въз основа на участието им в експеримента, събирането на данни, анализа и качеството на техните презентации.

Домашна работа

Задайте Домашна работа, която изисква от студентите да изследват и представят реално приложение на топлопроводимостта в инженерството или Материали: наука.

Заключение

Завършете урока, като обобщите научените ключови понятия и затвърдите значението на разбирането на топлопроводимостта във всеки живот и технологични приложения.

FROM GAME TO STEM

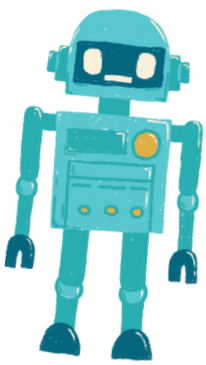
Co-funded by the Erasmus+ Programme of the European Union

Урок 8
Урок по биология на Ардуино

Изследване на фотосинтезата и растежа на растенията с Ардуино

Продължителност:

3 периода



Обективен:

- Студентите ще разберат процеса на фотосинтеза и значението му за растежа на растенията.
- Студентите ще проектират и проведат експеримент, за да изследват ефекта на факторите на околната среда върху фотосинтезата.
- Студентите ще програмират базирана на Ардуино система за наблюдение и [illegible]ане на данни, свързани с [illegible]ежа на растенията.

Материали::

- Ардуино дъски (по една на ученик или група)
- Сензори (напр. сензор за светлина, сензор за температура, сензор за влажност)
- LED лампи за отглеждане (по избор)
- Саксийни растения или семена
- Почва и съдове за засаждане
- Джъмперни проводници
- USB кабели за програмиране
- Компютри с инсталиран Ардуино IDE
- Проектор или бяла дъска за демонстрации
- Почва за саксии, вода и други Материали: за грижа за растенията

Занимания

Ден 1

<u>Въведение в Фотосинтезата (15 минути):</u>

- Започнете урока с представянето на концепцията за фотосинтезата и нейното значение за растежа на растенията.
- Обсъдете химичното уравнение на фотосинтезата и ролята на светлината, въглеродния диоксид и вода в процеса.

Фактори в Средата и Растежът на Растенията (30 минути):

- Обяснете как различни фактори от околната среда, като интензитет на светлината, температурата и влажността, могат да влияят върху фотосинтезата и растежа на растенията.
- Обсъдете защо тези фактори са важни за здравия развой на растенията.
- Подчертайте необходимостта от контролирани експерименти за изучаване на тези фактори.

<u>Въведение в Ардуино и Сензорите (20 минути):</u>

- Представете Ардуино като платформа за събиране на данни и обяснете нейните компоненти (микроконтролер, сензори).
- Покажете примери на сензори, често използвани в околната среда и грижата за растенията.
- Обяснете ролята на сензорите при събирането на данни за експериментите.
-

<u>Подготовка за Практическа Дейност (45 минути):</u>

- Предоставете на учениците Ардуино платки, сензори (например, сензор за светлина, температурен сензор, сензор за влажност) и LED растителни лампи (по избор).
- Наставете учениците да измислят и планират своите експерименти за растежа на растенията, фокусирайки се върху един околен фактор.
- Поискайте от учениците да подготвят саксии с почва и да засадят семена или малки растения в тях.

Ден 2

Настройка на експеримента и събиране на данни (60 минути):

- Започнете втория Ден, като позволите на учениците да организират свои експерименти.
- Учениците трябва да поставят сензорите в растителната среда, да ги свържат с Ардуино и да позиционират LED светлините за отглеждане (ако се използват).
- Инструктирайте учениците да събират данни, свързани с избрания фактор на околната среда, като интензитет на светлината или температура.
- Подчертайте важността на точното и редовно записване на данни.

Анализ на данни и дискусия (30 минути):

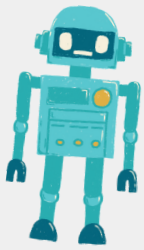
- Насочвайте учениците да анализират своите данни, търсейки тенденции или модели, свързани с растежа на растенията и избрания фактор.
- Обсъдете влиянието на фактора върху фотосинтезата и развитието на растенията.

Ден 3

Програмиране на системата за наблюдение (60 минути):

- Инструктирайте учениците да напишат код на Ардуино за наблюдение и събиране на данни от сензорите.
- Учениците трябва да програмират Ардуино да записва данни на определени интервали (напр. всеки час).

Ето пример за код на Ардуино за проста система за мониторинг на околната среда, която измерва и регистрира данни, свързани с интензитета на светлината, използвайки сензор за светлина. Можете да адаптирате този код за други фактори на околната среда, ако е необходимо:



```
#include <Wire.h>
#include <Adafruit_Sensor.h>
#include <Adafruit_TSL2591.h>
Adafruit_TSL2591 tsl = Adafruit_TSL2591(2591);
void setup() {
 Serial.begin(9600);
  // Initialize the light sensor
 if(!tsl.begin()) {
   Serial.println("Light sensor not found. Check wiring.");
   while(1);
 }
 tsl.setGain(TSL2591_GAIN_LOW);    // Adjust the gain (options: LOW, MED, HIGH, MAX)
 tsl.setTiming(TSL2591_INTEGRATIONTIME_100MS); // Adjust the integration time (options:
100, 200, 300, 400, 500, 600)
}
void loop() {
 // Read and print light intensity data
 uint16_t luminance = tsl.getLuminosity(TSL2591_VISIBLE);
 Serial.print("Light Intensity (Lux): ");
 Serial.println(luminance);
    // Add code here to log data to an SD card, display on an LCD, or transmit to a
computer/server.
 // Wait for a specific time interval (e.g., 1 hour)
 delay(3600000); // Adjust the delay time as needed
}
```

В този код:

Ние използваме библиотеката Adafruit TSL2591 за взаимодействие със сензора за светлина TSL2591. Уверете се, че имате инсталирана тази библиотека във вашата Ардуино IDE.
Функцията setup() инициализира серийната комуникация за извеждане на данни и настройва светлинния сензор. Той също така конфигурира усилването и времето за интегриране на сензора въз основа на вашите изисквания.
Във функцията loop() програмата непрекъснато чете данните за интензитета на светлината (в луксове) от сензора, използвайки tsl.getLuminosity(TSL2591_VISIBLE).
Можете да добавите код в рамките на цикъла, за да регистрирате данните на SD карта, да ги покажете на LCD екран или да ги прехвърлите на компютър или сървър за допълнителен анализ. Например, можете да използвате модул за SD карта, за да съхранявате данни локално.
Закъснението в края на цикъла е настроено да изчаква определен интервал от време (напр. 1 час), преди да вземе следващото отчитане. Можете да регулирате времето на забавяне, за да контролирате честотата на събиране на данни.

Този код предоставя основна рамка за наблюдение на интензитета на светлината и можете да я разширите, като добавите още сензори за наблюдение на допълнителни фактори на околната среда, като температура и влажност. Не забравяйте да коригирате кода, за да съответства на конкретните сензори и хардуер, които използвате в експеримента си.

Представяне на проекта и дискусия (60 минути):

- Всяка група представя своята настройка, методология и резултати от експеримента за растеж на растения пред класа.
- Насърчете учениците да обяснят своите наблюдения и последиците от техните открития за грижата за растенията и селското стопанство.
- Улеснете дискусия в клас за значението на фотосинтезата в екосистемата и как технологията (Ардуино) може да помогне в научните изследвания.

Оценки

Оценявайте учениците въз основа на участието им в експеримента, събирането на данни, анализа и качеството на техните презентации.

Домашна работа

Задайте Домашна работа, която изисква от студентите да изследват и представят реално приложение на фотосинтеза и мониторинг на околната среда в селското стопанство или екологията.

Заключение

Завършете урока, като обобщите научените ключови концепции от биологията и затвърдите значението на фотосинтезата и факторите на околната среда за растежа на растенията. Подчертайте ролята на технологията за напредъка на научното разбиране.

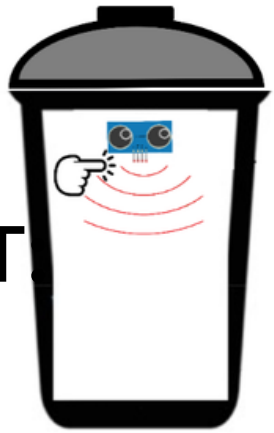
FROM GAME TO STEM

Co-funded by the Erasmus+ Programme of the European Union

Урок 9

Урок по екология на Ардуино

Изграждане на интелигентна кофа за боклук с Ардуино



Продължителност

3 периода

Обективен

- Студентите ще разберат основите на програмирането на Ардуино и неговите приложения в автоматизацията.
- Студентите ще научат за ултразвуковите сензори и как могат да се използват за откриване на обекти.
- Учениците ще създадат прототип на Smart Dustbin и ще го програмират да отваря капака си, когато бъде открит обект.
- Студентите ще изследват екологичните ползи от интелигентните системи за управление на отпадъците.

Материали

- Ардуино Uno дъски (по една на ученик или група)
- Ултразвуков сензор HC-SR04 (един на ученик или група)
- Сервомотори (по един на ученик или група)
- Бредове и джъмпери
- Малка картонена кутия или контейнер (за кофата за боклук)
- Картонено или пластмасово капаче (за симулиране на капака на кофата за боклук)
- USB кабели за програмиране
- Компютри с инсталиран Ардуино IDE
- Проектор или бяла дъска за демонстрации

Занимания

Ден 1

<u>Въведение в Ардуино и Електроника (15 минути):</u>

- Началото на урока с представяне на Ардуино като микроконтролерна платформа, използвана за различни проекти.
- Обсъждане на значението на електрониката и автоматизацията в съвременните технологии.

<u>Ултразвукови Сензори и Откриване на Обекти (30 минути):</u>

- Обяснение на принципа на ултразвуковите сензори и как работят за измерване на разстояния.
- Обсъждане на компонентите на ултразвуковия сензор HC-SR04 (предавател и приемник).
- Илюстриране на това как ултразвуковите сензори могат да бъдат използвани за откриване на обекти и измерване на разстояния.

<u>Въведение в Сервомоторите (20 минути):</u>

- Представяне на сервомоторите и тяхното приложение за контрол на механични движения.
- Показване на примери как сервомоторите могат да бъдат използвани в проекти като отваряне на капак.

<u>Настройка на Практическата Дейност (45 минути):</u>

- Предоставяне на всяка група с Ардуино Uno, ултразвуков сензор, сервомотор, макетна платка и жички с щипки.
- Инструктиране на студентите да сглобят прототипа на Умната Кофа, разполагайки ултразвуковия сензор и сервомотор на подходящи места.

Ден 2

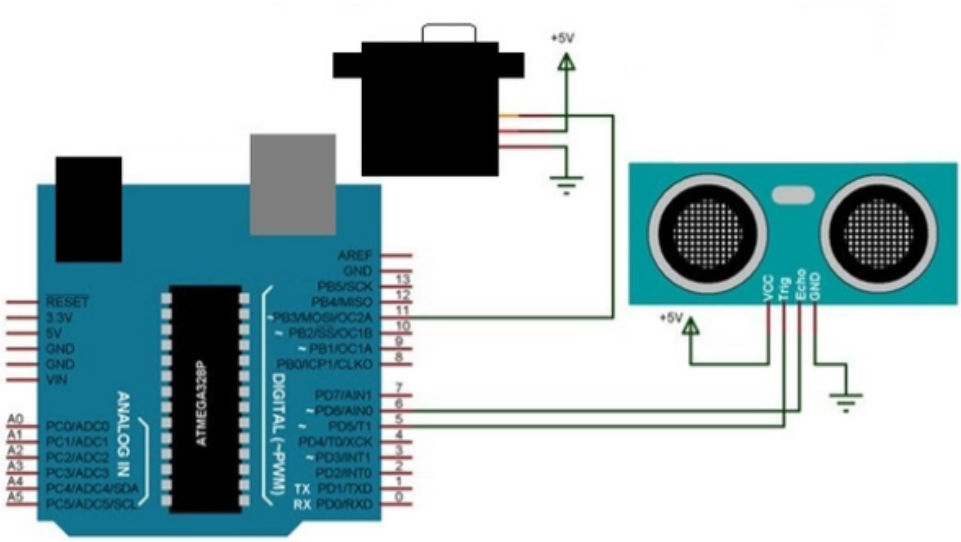
Основи на програмирането на Ардуино (30 минути):

- Научете учениците на основите на програмирането на Ардуино , включително setup(), loop() и pinMode().
- Дайте примери за основен код на Ардуино за мигане на светодиода.

Програмиране на интелигентната кофа за боклук (60 минути):

- Насочете учениците да напишат код на Ардуино за управление на Smart Dustbin въз основа на показанията на ултразвуков сензор.
- Обяснете логиката за отваряне на капака, когато обект бъде открит в определен диапазон.

ШЕМА:



Ето примерен код на Ардуино за интелигентна кофа за боклук, използваща ултразвуков сензор (HC-SR04) и сервомотор. Този код позволява на сервомотора да отвори капака на кофата за боклук, когато бъде открит обект в определен диапазон:

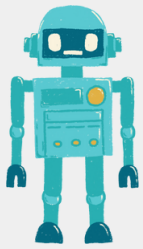
```
#include <Servo.h>

#define TRIGGER_PIN 9
#define ECHO_PIN 10
#define SERVO_PIN 11

Servo myservo; // Create a Servo object
int distance; // Variable to store distance measured by the Ultrasonic sensor

void setup() {
 myservo.attach(SERVO_PIN); // Attach the Servo to the specified pin
 pinMode(TRIGGER_PIN, OUTPUT);
 pinMode(ECHO_PIN, INPUT);
 Serial.begin(9600); // Initialize serial communication for debugging
}
```

```
void loop() {
 // Send a brief pulse to trigger the Ultrasonic sensor
 digitalWrite(TRIGGER_PIN, LOW);
 delayMicroseconds(2);
 digitalWrite(TRIGGER_PIN, HIGH);
 delayMicroseconds(10);
 digitalWrite(TRIGGER_PIN, LOW);
 // Read the duration of the echo pulse and calculate the distance
 duration = pulseIn(ECHO_PIN, HIGH);
 distance = (duration / 2) / 29.1; // Calculate distance in centimeters
 // Check if an object is within the specified range (adjust as needed)
 if (distance < 20) { // You can adjust the distance threshold here
  // If an object is detected, open the flap
   myservo.write(90); // Rotate the Servo to open the flap (adjust the angle as needed)
  delay(1000); // Wait for 1 second
  myservo.write(0); // Rotate the Servo back to close the flap
 }
 // Print the distance to the Serial Monitor for debugging
 Serial.print("Distance: ");
 Serial.print(distance);
 Serial.println(" cm");
 // Add a delay between readings to prevent rapid triggering
 delay(1000); // You can adjust the delay time as needed
}
```



В този код:
Включваме серво библиотеката и дефинираме номерата на щифтовете за тригера и ехото на ултразвуковия сензор, както и контролния щифт на сервомотора.
Във функцията setup() прикрепяме сервомотора към посочения щифт и конфигурираме щифта на тригера като изход, а щифта за ехо като вход. Ние също инициализираме серийна комуникация за отстраняване на грешки.
Функцията loop() многократно изпълнява следните стъпки:
Изпраща кратък импулс към ултразвуковия сензор, за да задейства измерване на разстоянието.
Измерва продължителността на ехо импулса и изчислява разстоянието в сантиметри.
Проверява дали обектът е в определения диапазон (20 см в този пример) и ако е така, отваря капака на кофата за боклук чрез завъртане на сервомотора до определен ъгъл (90 градуса).
Отпечатва разстоянието до серийния монитор за целите на отстраняване на грешки.
Добавя забавяне между показанията, за да предотврати бързо задействане.
Можете да регулирате прага на разстоянието, ъгъла на сервомотора и времената на забавяне, за да отговарят на вашите специфични настройки и изисквания.

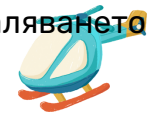
Тестване и отстраняване на неизправности (30 минути):

- Накарайте учениците да тестват своите прототипи на Smart Dustbin и да отстранят всички проблеми с кода или хардуера.
- Насърчавайте експериментирането с различни разстояния на откриване и ъгли на серво мотора.

Ден 3

Представяне на проекта и дискусия (60 минути):

- Всяка група представя своя проект за Smart Dustbin пред класа, като обяснява дизайна, компонентите и кода.
- Обсъдете ползите за околната среда от интелигентните системи за управление на отпадъците и тяхното потенциално въздействие върху намаляването на отпадъците.



Отворена дискусия и бъдещи подобрения (30 минути):

- Улеснете дискусия в клас относно потенциални подобрения на Smart Dustbin и други приложения на подобна технология.
- Насърчете учениците да обмислят идеи за по-нататъшно подобряване на решенията за управление на отпадъците.

Оценки

Оценявайте учениците въз основа на тяхното участие, функционалност на проекта, разбиране на програмирането на Ардуино и способността им да отстраняват проблеми.

Домашна работа

Задайте Домашна работа, която изисква от студентите да проучат и представят примери от реалния свят на интелигентни системи за управление на отпадъците и тяхното въздействие върху устойчивостта.

Заключение

Завършете урока, като обобщите научените ключови концепции, като подчертаете пресечната точка на технологиите и екологичните решения и насърчите учениците да мислят критично за прилагането на технологиите за положителна промяна.

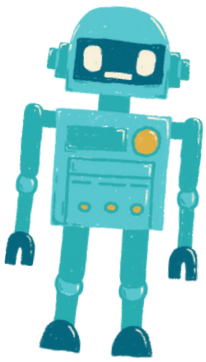
FROM GAME TO STEM

Co-funded by the Erasmus+ Programme of the European Union

Урок 10
Ардуино Урок по екология

Изследване на качеството на въздуха с Ардуино

Продължителност:
1 периоди



Обективен

- Учениците ще научат за значението на качеството на въздуха за здравето на околната среда.
- Студентите ще разберат как замърсителите на въздуха могат да повлияят на екосистемите и човешкото здраве.
- Студентите ще създадат прост базиран на Ардуино сензор за качество на въздуха и ще събират данни.
- Студентите ще обсъдят значението на мониторинга на качеството на въздуха за екологичното благосъстояние.

Материали

- Ардуино Uno дъски (по една на ученик или група)
- Сензорен модул за качество на въздуха (напр. MQ-135)
- Джъмперни проводници
- USB кабели за програмиране
- Компютри с инсталирано Arduino IDE
- Проектор или бяла дъска за демонстрации

Занимания

Ден 1

<u>Въведение в качеството на въздуха (10 минути):</u>

- Започнете урока, като обсъдите значението на чистия въздух и неговото въздействие върху екосистемите.
- Обяснете как замърсителите на въздуха могат да повлияят както на околната среда, така и на човешкото здраве.

<u>Сензори за качество на въздуха (20 минути):</u>

- Представете сензорите за качество на въздуха и тяхната роля в мониторинга на замърсяването на въздуха.
- Обяснете основната работа на сензорите за качество на въздуха и как те измерват различни газове.

<u>Основи на Ардуино (10 минути):</u>

- Представете Ардуино като микроконтролерна платформа за събиране на данни.
- Покажете на учениците компонентите на дъска Arduino и основите на писане и качване на код.

<u>Подготовка за практическа дейност (15 минути):</u>

- Осигурете на всяка група Arduino Uno, сензорен модул за качество на въздуха и джъмпери.
- Инструктирайте учениците да сглобят сензора за качество на въздуха и да го свържат към Ардуино.

<u>Изграждане и програмиране на сензора за качество на въздуха (20 минути):</u>

- Насочете учениците в изграждането на тяхната сензорна система за качество на въздуха, базирана на Ардуино.
- Покажете на учениците как да напишат код на Ардуино, за да събират данни за качеството на въздуха от сензора.

Ето прост пример за код на Arduino за наблюдение на качеството на въздуха с помощта на сензор за качество на въздуха MQ-135. Този код чете данни от сензора и ги показва на серийния монитор. Уверете се, че имате подходящите библиотеки, инсталирани във вашето Arduino IDE, тъй като сензорът MQ-135 може да изисква калибриране за точни резултати.

```
// Include necessary libraries
#include <Wire.h>
#include <Adafruit_Sensor.h>
#include <Adafruit_MQ135.h>

// Define the analog pin where the MQ-135 sensor is connected
#define MQ135_PIN A0

// Create an instance of the Adafruit MQ135 class
Adafruit_MQ135 mq135(MQ135_PIN);

void setup() {
 // Initialize serial communication for data output
 Serial.begin(9600);
}

void loop() {
 // Read the MQ-135 sensor's data
 float airQuality = mq135.readCO2();

 // Print the air quality data to the Serial Monitor
 Serial.print("Air Quality: ");
 Serial.print(airQuality);
 Serial.println(" ppm");

 // Add a delay before the next reading
 delay(2000); // Adjust the delay time as needed
}
```

В този код:
Включваме необходимите библиотеки, включително библиотеките Adafruit Sensor и Adafruit MQ135, които обикновено се използват за работа със сензора за качество на въздуха MQ-135. Уверете се, че имате инсталирани тези библиотеки във вашето IDE на Arduino.
Ние дефинираме аналоговия щифт (A0 в този пример), където сензорът MQ-135 е свързан към Ардуино.
Създава се екземпляр на класа Adafruit_MQ135 за взаимодействие със сензора.
Във функцията setup() инициализираме серийна комуникация за извеждане на данни към серийния монитор.
Във функцията loop() ние непрекъснато четем данните за качеството на въздуха от сензора MQ-135, използвайки mq135.readCO2(). Данните представляват концентрацията на CO2 в части на милион (ppm).
Данните за качеството на въздуха се отпечатват на серийния монитор и има закъснение от 2 секунди (регулируемо), преди да се вземе следващото отчитане.
Моля, обърнете внимание, че сензорът MQ-135 може да изисква калибриране за точни показания и предоставеният тук код служи като основен пример. В зависимост от вашето конкретно приложение и изисквания за калибриране, може да се наложи да коригирате съответно кода и процеса на калибриране.

Събиране на данни и дискусия (20 минути):

- Инструктирайте учениците да събират данни за качеството на въздуха, като пуснат своите сензори в различни среди (напр. на закрито, близо до път, в градина).
- Накарайте учениците да записват и споделят своите данни с класа.
- Водете дискусия в клас относно разликите в данните за качеството на въздуха и потенциалните последици за околната среда.

Оценки

Оценявайте учениците въз основа на тяхното участие, събиране на данни и способността им да обсъждат въздействието на качеството на въздуха върху екологичните системи.

Домашна работа

Задайте Домашна работа, която изисква от учениците да изследват и представят примери от реалния свят на екологични проблеми, свързани със замърсяването на въздуха и значението на мониторинга на качеството на въздуха за смекчаване на тези проблеми.

Заключение

Завършете урока, като обобщите научените ключови екологични понятия, като подчертаете ролята на технологията (Ардуино) в мониторинга на околната среда и насърчите учениците да обмислят значението на качеството на въздуха за екологичното благополучие.

FROM GAME TO STEM

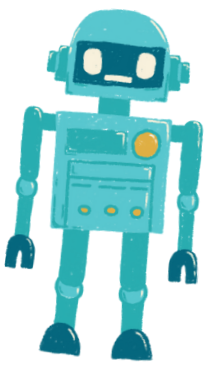
With the support of the Erasmus+ Programme of the European Union

Ders 1
Arduino Matematik Dersi

Arduino ile Trigonometri

Süre:

3 dönem



Hedef:

- Öğrenciler temel trigonometri kavramlarını, sine (sinüs), cosine (kosinüs) ve tangent (tanjant) dahil olmak üzere, anlayacaklar.
- Öğrenciler trigonometrik fonksiyonları gerçek dünya problemlerini çözmek için uygulayacaklar.
- Öğrenciler, bir servo motor kullanarak basit bir açı ölçüm cihazı oluşturmak için bir Arduino programlayacaklar

Malzemeler::

- Arduino panoları (her öğrenci veya grup için bir tane)
- Servo motorlar
- Breadboardlar (deneme tahtaları)
- Bağlantı telleri (Jumper teller)
- Goniometreler (Açı ölçerler)
- Cetveller (Rulers)
- Programlama için USB kabloları
- Arduino IDE yüklü bilgisayarlar
- Gösterimler için projektör veya beyaz tahta

Etkinlikler:

Gün 1

Giriş Trigonometri (15 dakika):

- Dersi, trigonometri kavramını ve gerçek dünya uygulamalarındaki önemini tanıtarak başlatın.
- Sinüs, kosinüs ve tanjantı, dik üçgenlerdeki kenarların oranları olarak kısaca açıklayın.
- Trigonometrik fonksiyonların açılar ve mesafelerle ilgili problemleri nasıl çözmek için kullanılabileceğini tartışın.

Trigonometri Temelleri (30 dakika):

- Sinüs, kosinüs ve tanjantın tanımlarına daha detaylı bir şekilde girin.
- Bu fonksiyonların nasıl kullanıldığını, dik üçgenlerde eksik açıları veya kenar uzunluklarını nasıl bulmak için kullanıldığına dair örnekler verin.
- Öğrencilere kağıt üzerinde temel trigonometri problemlerini çözmeyi pratiğe bırakın.

Arduino ve Servo Motorlarına Giriş (20 dakika):

- Arduino'yu, aletler (microcontroller, sensörler, aktuatörler) oluşturmak için bir platform olarak tanıtın.
- Servo motor kavramını ve belirli açılara hareket etmelerini nasıl kontrol edebileceğinizi açıklayın.
- Arduino ile servo motor kontrolünün örneklerini gösterin.

Elle Yapılacak Aktivite (45 dakika):

- Öğrencileri çiftlere veya küçük gruplara ayırın.
- Her gruba bir Arduino kartı, bir servo motor, bir breadboard ve bağlantı telleri verin.
- Öğrencilere, bir referans olarak bir açı ölçer kullanarak basit bir servo motor açı ölçüm cihazı monte etmelerini söyleyin.
- Öğrencilere, kullanıcı girişine dayalı olarak servo motoru belirli bir açıya taşımak için Arduino kodu yazmada rehberlik edin.

Gün 2

Trigonometri Kavramlarının Gözden Geçirilmesi (20 dakika):

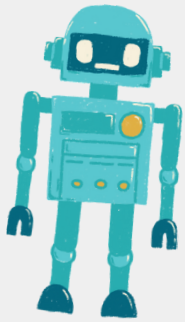
- İkinci Gün'e, İkinci Gün'de öğrenilen sinüs, kosinüs ve tanjant kavramlarını gözden geçirerek başlayın.
- Öğrencilerin trigonometrik fonksiyonları kullanarak çözmeleri için ek pratik problemler sunun.

Açı Ölçüm Cihazını Programlama (45 dakika):

- İlk Gün'den gelen elle yapılacak aktiviteye devam edin.
- Öğrencilere, servo motoru ve bir sayısal ekranı (örneğin, Seri Monitör) kullanarak açıları doğru bir şekilde ölçmek ve görüntülemek için Arduino kodlarını tamamlamalarını söyleyin.
- Öğrencileri, servonun pozisyonunu açılara dönüştürmek için trigonometriyi uygulamaya teşvik edin.

İşte bir servo motor kullanarak basit bir açı ölçüm cihazı oluşturmak için kullanabileceğiniz bir Arduino kodu örneği. Bu kod, kullanıcı girişine dayalı olarak servo motoru belirli bir açıya taşır ve ölçülen açıyı Seri Monitör üzerinde görüntüler.

```
#include <Servo.h>
Servo myservo; // Create a Servo object
void setup() {
myservo.attach(9); // Attaches the servo to digital pin 9
Serial.begin(9600); // Initialize serial communication for debugging
}
void loop() {
int angle; // Variable to store the desired angle
Serial.println("Enter an angle (0-180): ");
while (!Serial.available()) {
  // Wait for user input
}
angle = Serial.parseInt(); // Read the angle entered by the user
if (angle >= 0 && angle <= 180) {
// Check if the entered angle is within the valid range
myservo.write(angle); // Move the servo to the specified angle
delay(500); // Delay for stability
Serial.print("Measured angle: ");
Serial.print(angle);
Serial.println(" degrees");
} else {
Serial.println("Invalid angle. Please enter a value between 0 and 180.");
}
}
```



Bu kodda:

1. Servo motoru kontrol etmek için Servo kütüphanesini dahil ediyoruz.
2. setup() fonksiyonunda, servoyu dijital pin 9'a bağlıyor ve hata ayıklama için seri iletişimi başlatıyoruz.
3. loop() fonksiyonunda, kullanıcının Seri Monitör aracılığıyla istediği açıyı girmesini bekliyoruz. Kullanıcıdan 0 ila 180 derece arasında bir açı girmesi istenir.
4. Girilen açının geçerli aralıkta olup olmadığını kontrol ediyoruz (0 ila 180 derece). Geçerliyse, servoyu belirtilen açıya taşıyoruz (myservo.write(angle)) ve ölçülen açıyı Seri Monitör üzerinde görüntülüyoruz.
5. Eğer girilen açı geçerli aralık dışındaysa, bir hata mesajı görüntülüyoruz.

Bu kodu öğrencilerinizle kullanmak için, servo motorunun Arduino'daki dijital pin 9'a bağlı olduğundan ve Seri Monitör'ü açtıklarından emin olun. Açıları girmelerini sağlayın ve ölçülen açıları gözlemleyin.

Gün 3

Proje Sunumu ve Tartışma (60 dakika):

Her grup, Arduino açı ölçüm cihazlarını sınıfa sunar.
Öğrencilere projelerinde trigonometri kavramlarını nasıl uyguladıklarını açıklamalarını teşvik edin.
Açı ölçüm cihazlarının ve trigonometrinin gerçek dünya uygulamalarını tartışın.
Projede karşılaşılan zorluklar ve çözümler hakkında sınıf içi bir tartışma düzenleyin.

Değerlendirmeler

Öğrenciler, katılımları, kod kaliteleri, açı ölçümünün doğruluğu ve cihazlarının arkasındaki trigonometrik prensipleri açıklama yeteneklerine göre değerlendirilecektir.

Ödev

Öğrencilere, trigonometrinin mühendislik, fizik veya mimarlık gibi çeşitli alanlarda gerçek dünya uygulamalarını araştırmalarını ve sunmalarını isteyebilirsiniz.

Sonuç

Dersi, öğrenilen temel trigonometrik kavramları özetleyerek ve bu kavramların Arduino cihaz geliştirmesi gibi pratik kullanımlarını vurgulayarak sonlandırabiliriz. Matematik ve teknoloji arasındaki bağlantıyı ön plana çıkarmak oldukça önemlidir.

FROM GAME TO STEM

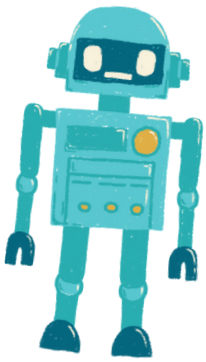
With the support of the Erasmus+ Programme of the European Union

Ders 2
Arduino Matematik Dersi

Aljebra Modelleme Arduino ile

Süre:

2 ders



Hedef:

- Öğrenciler, cebirsel kavramları gerçek dünya problemlerini çözmek için nasıl kullanacaklarını öğrenecekler.
- Öğrenciler, matematiksel modelleme ve programlama becerilerini kullanarak pratik bir sorunu çözen bir Arduino cihazı oluşturacaklar.
- Öğrenciler, değişkenler ve denklemlerle deneyim kazanacaklar.

Malzemeler::

- Arduino panoları (her öğrenci veya grup için bir tane)
- Breadboardlar (deneme tahtaları)
- Sensörler veya giriş cihazları (örneğ sıcaklık sensörü, ışık sensörü, bas düğmesi)
- LED'ler, dirençler ve bağlantı telleri
- Programlama için USB kabloları
- Arduino IDE yüklü bilgisayarlar
- Gösterimler için projektör veya beyaz tahta

Etkinlikler:

Gün 1

Aljebra Modellemesine Giriş (15 dakika):

- Dersi, aljebra modellemeyi ve gerçek dünya problemlerini çözmedeki önemini tanıtarak başlatın.
- Matematiksel modellemede değişkenlerin ve denklemlerin önemini tartışın.

Değişkenler ve Denklemler (30 dakika):

- Değişkenlerin konseptini gözden geçirin ve bilinmeyen değerleri temsil etmek için nasıl kullanıldıklarını açıklayın.
- Değişkenler arasındaki ilişkileri modelleme yolu olarak doğrusal denklemleri (örneğin, y=mx+b) tanıtın.
- Denklemler kullanılarak temsil edilebilecek gerçek dünya problemlerine dair örnekler sunun.

Arduino Temelleri (20 dakika):

- Arduino'yu bir cihaz oluşturma platformu olarak tanıtın ve bileşenlerini açıklayın.
- Basit Arduino projelerinin örneklerini gösterin ve programlamada değişkenlerin nasıl kullanılabileceğini açıklayın.

Elle Yapılacak Aktivite (45 dakika):

- Öğrencileri çiftlere veya küçük gruplara ayırın.
- Her gruba bir Arduino kartı, bir deneme tahtası, bir sensör veya giriş cihazı (örneğin, sıcaklık sensörü) ve bir LED verin.
- Öğrencilere, bir sensörü veri okumak ve bir LED'i veriyi görsel olarak temsil etmek için kullanacak basit bir Arduino projesi oluşturmalarını söyleyin.
- Öğrencileri sensör okumalarını saklamak için değişkenleri kullanmaya ve sensör verilerine dayalı olarak LED'i kontrol etmek için denklemler oluşturmaya teşvik edin.
- Gadget'larının kullanışlı olabileceği farklı gerçek dünya senaryolarını tartışın.

Gün 2

Aljebra Kavramlarının Gözden Geçirilmesi (20 dakika):

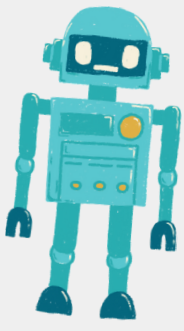
- İkinci Gün'e, değişkenler, denklemler ve matematiksel modelleme kavramları gözden geçirerek başlayın.
- Önceki günün Arduino projelerini ve uygulamalarını tartışın.

Aljebra Modelini Programlama (45 dakika):

- İlk Gün'den gelen elle yapılacak aktiviteye devam edin.
- Öğrencilere, Arduino kodlarını değiştirerek sensör verilerini kullanarak matematiksel bir model oluşturmalarını söyleyin.
- Farklı denklemler ve değişkenlerle deneysel çalışmalar yaparak sensör verileri ile LED çıkışı arasındaki ilişkiyi temsil etmelerini teşvik edin.
- Modelin tahminlerde bulunmak veya gerçek dünya sistemlerini kontrol etmek için nasıl kullanılabileceğini tartışın.

İşte basit bir aljebra modelleme projesi için Arduino kodu örneği. Bu projede öğrenciler, sıcaklık sensörü kullanarak sıcaklık verilerini okuyacaklar ve sıcaklık okumasına bağlı olarak bir LED'i kontrol edecekler. Kod, LED'in ne zaman açılması veya kapanması gerektiğini belirlemek için değişkenler ve bir denklem kullanır.

```
// Define the pins for the temperature sensor, LED, and a resistor (if needed)
const int temperatureSensorPin = A0; // Analog pin for the temperature sensor
const int ledPin = 13; // Use the built-in LED on most Arduino boards
const float thresholdTemperature = 25.0; // Define the threshold temperature
void setup() {
pinMode(ledPin, OUTPUT); // Set the LED pin as an output
Serial.begin(9600); // Initialize serial communication for debugging (optional)
}
void loop() {
// Read the temperature from the sensor
int sensorValue = analogRead(temperatureSensorPin);
// Convert the analog value to temperature in degrees Celsius
float temperatureCelsius = map(sensorValue, 0, 1023, 0, 100); // Adjust the mapping as
needed
// Check if the temperature is above the threshold
if (temperatureCelsius > thresholdTemperature) {
digitalWrite(ledPin, HIGH); // Turn the LED on
} else {
digitalWrite(ledPin, LOW); // Turn the LED off
}
// Print the temperature to the serial monitor for debugging (optional)
Serial.print("Temperature: ");
Serial.print(temperatureCelsius);
Serial.println(" °C");
// Delay for a moment to avoid rapid LED toggling
delay(1000); // Delay for 1 second
}
```



Bu kodda:

- Sıcaklık sensörü için pinler (bir analog pinde bağlı), LED için pinler (çoğu Arduino kartında bulunan yerleşik LED) ve LED'in ne zaman açılması gerektiğini belirleyen bir eşik sıcaklık tanımlarız.
- setup() fonksiyonunda, LED pinini çıkış olarak ayarlarız ve hata ayıklama için (isteğe bağlı) seri iletişimi başlatırız.
- loop() fonksiyonunda, sensörden sıcaklık değerini analogRead() ile okuruz. Sensörün özelliklerine dayanarak analog sensör değerini derece Celsius cinsinden sıcaklığa çeviririz.
- Ardından, sıcaklığın tanımlanan eşik sıcaklığı (thresholdTemperature) aşıp aşılmadığını kontrol ederiz. Eğer sıcaklık eşik sıcaklığın üstündeyse, LED açılır; aksi halde kapanır.
- İsteğe bağlı: Hata ayıklama amaçları için sıcaklığı Seri Monitör'e yazdırırız.

Bu kodu kullanmak için öğrenciler, bir sıcaklık sensörünü (örneğin, LM35) Arduino'daki bir analog pine ve bir LED'i bir dijital pine bağlamaları gerekecektir. Kod, sensörden sıcaklık okur ve sıcaklık okumasına ve eşik değerine dayalı olarak LED'i kontrol eder.

Proje Sunumu ve Tartışma (30 dakika):

- Her grup, Arduino cihazlarını sınıfa sunar.
- Öğrencilere oluşturdukları matematiksel modeli ve nasıl çalıştığını açıklamalarını teşvik edin.
- Aljebra modellemenin pratik problemleri çözmedeki uygulamalarını tartışmak için bir sınıf tartışması düzenleyin.

Değerlendirmeler

Öğrencilerin katılımı, Arduino cihazlarının işlevselliği ve projelerinde kullandıkları aljebra kavramlarını açıklama becerilerine dayalı olarak öğrencileri değerlendirebilirsiniz.

Ödev

Öğrencilere, aljebra modelleme ve Arduino cihazı kullanarak çözülebilecek bir gerçek dünya problemi araştırmalarını ve sunmalarını isteyebilirsiniz.

Sonuç

Dersi, öğrenilen temel aljebra kavramlarını özetleyerek ve matematiksel modellemenin teknoloji ve mühendislikteki önemini vurgulayarak sonlandırabiliriz.

FROM GAME TO STEM

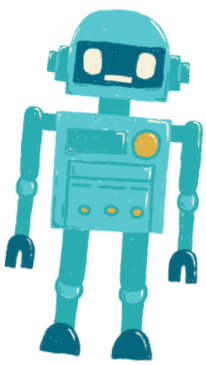
Co-funded by the Erasmus+ Programme of the European Union

Lección 3
Lección de Física con Arduino

Explorando el Movimiento y la Aceleración con Arduino

Duración:

3 períodos



Objetivo:

- Los estudiantes comprenderán los principios básicos del movimiento y la aceleración.
- Los estudiantes aplicarán ecuaciones cinemáticas para resolver problemas del mundo real relacionados con el movimiento.
- Los estudiantes programarán un Arduino para medir y mostrar la aceleración utilizando un sensor.

Materiales::

- Placas Arduino (una por estudiante o grupo)
- Módulos de sensor de acelerómetro (por ejemplo, ADXL345)
- Protoboards
- Cables de puente
- Cables USB para la programación
- Computadoras con el entorno de desarrollo Arduino IDE instalado
- Proyector o pizarra blanca para las demostraciones
- Opcional: Objetos de la vida real para experimentos de movimiento (por ejemplo, autos de juguete, pelotas)

Actividades

Día 1

Introducción al Movimiento y la Aceleración (15 minutos):

- Comienza la lección presentando el concepto de movimiento y aceleración.
- Define términos clave como velocidad, aceleración y desaceleración.
- Discute la importancia de comprender el movimiento en diversos campos, como la física, la ingeniería y el transporte.

Física del Movimiento (30 minutos):

- Explica las ecuaciones del movimiento, incluyendo:
- Desplazamiento: $s = ut + \frac{1}{2}at^2$
- Velocidad: $v = u + at$
- Aceleración: $a = \frac{v-u}{t}$
- Proporciona ejemplos de cómo se utilizan estas ecuaciones para analizar el movimiento.
- Realiza experimentos simples de movimiento (por ejemplo, rodar una pelota cuesta abajo) para demostrar los principios del movimiento.

Introducción a Arduino y Acelerómetros (20 minutos):

- Presenta Arduino como una plataforma para crear dispositivos y explica sus componentes.
- Explica el concepto de acelerómetros y cómo miden la aceleración.
- Muestra ejemplos de datos de acelerómetros y explica cómo se relacionan con el movimiento del mundo real.

Actividad Práctica (45 minutos):

- Divide a los estudiantes en parejas o grupos pequeños.
- Proporciona a cada grupo una placa Arduino, un módulo de sensor de acelerómetro, una protoboard y cables de puente.
- Instruye a los estudiantes para que ensamblen un circuito simple que conecte el acelerómetro a la placa Arduino.
- Guía a los estudiantes en la escritura de código Arduino para leer y mostrar datos de aceleración del sensor en el Monitor Serie.

Día 2

Revisión de las Ecuaciones Cinemáticas (20 minutos):

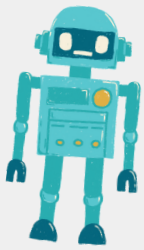
- Comienza el segundo día revisando las ecuaciones cinemáticas discutidas en el primer día.
- Proporciona ejemplos adicionales para que los estudiantes los resuelvan utilizando estas ecuaciones.

Programación del Dispositivo de Medición de Aceleración (45 minutos):

- Continúa con la actividad práctica iniciada en el primer día.
- Instruye a los estudiantes para que completen su código de Arduino y puedan leer y mostrar datos de aceleración en tiempo real del acelerómetro.
- Anima a los estudiantes a calibrar el sensor si es necesario y a aplicar las ecuaciones de aceleración para interpretar los datos.

Aquí tienes un ejemplo de código de Arduino para medir y mostrar la aceleración utilizando un sensor de acelerómetro ADXL345. Este código permitirá a tus estudiantes conectar el sensor de acelerómetro con Arduino y mostrar los valores de aceleración en el Monitor Serie.

```
#include <Wire.h>
#include <Adafruit_Sensor.h>
#include <Adafruit_ADXL345_U.h>
Adafruit_ADXL345_Unified accel = Adafruit_ADXL345_Unified(12345);
void setup(void) {
 Serial.begin(9600);
 Serial.println("Accelerometer Test"); Serial.println("");
  if(!accel.begin())
 {
/* There was a problem detecting the ADXL345 ... check your connections*/
   Serial.println("Ooops, no ADXL345 detected ... Check your wiring!");
   while(1);
 }
}
void loop(void) {
 sensors_event_t event;
 accel.getEvent(&event);
 /* Display the acceleration values (in m/s^2) */
 Serial.print("X: "); Serial.print(event.acceleration.x); Serial.print(" ");
 Serial.print("Y: "); Serial.print(event.acceleration.y); Serial.print(" ");
 Serial.print("Z: "); Serial.print(event.acceleration.z); Serial.println(" m/s^2");
 delay(500); // Delay for half a second between readings
}
```



En este código:
Incluimos las bibliotecas necesarias para el sensor de acelerómetro ADXL345.
Creamos un objeto Adafruit_ADXL345_Unified llamado "accel" para interactuar con el sensor.
En la función "setup()", inicializamos la comunicación serial para fines de depuración y verificamos si se detecta el sensor de acelerómetro. Si no se detecta, se mostrará un mensaje de error.
En la función "loop()", leemos continuamente los datos de aceleración del sensor utilizando "accel.getEvent(&event)" y mostramos los valores de aceleración en los ejes X, Y y Z en metros por segundo al cuadrado (m/s^2) en el Monitor Serie.
Para utilizar este código, asegúrate de que tus estudiantes hayan conectado correctamente el sensor de acelerómetro ADXL345 al Arduino. El sensor debe estar conectado a los pines apropiados (por ejemplo, SDA y SCL) para la comunicación I2C. Cuando el Arduino esté encendido y ejecutando este código, mostrará continuamente los datos de aceleración en el Monitor Serie.
Ten en cuenta que es posible que debas instalar la biblioteca Adafruit ADXL345 a través del Administrador de bibliotecas de Arduino para poder utilizar este código.

Presentación del Proyecto y Discusión (60 minutos):

- Cada grupo presenta su dispositivo de medición de aceleración basado en Arduino a la clase.
- Anima a los estudiantes a explicar cómo aplicaron las ecuaciones cinemáticas y los principios de la física en sus proyectos.
- Habla sobre las aplicaciones del mundo real de los dispositivos de medición de aceleración en diversos campos.
- Facilita una discusión en clase sobre los desafíos y soluciones encontrados durante el proyecto.

Evaluaciones

Evaluar a los estudiantes en función de su participación, calidad del código, precisión de la medición de aceleración y su capacidad para explicar los principios de la física detrás de su dispositivo.

Tarea

Asignar una tarea que requiera a los estudiantes investigar y presentar una aplicación del mundo real de la medición de aceleración en física o ingeniería.

Resumen

Concluyan la lección resumiendo los conceptos clave de física aprendidos y enfatizando su uso práctico en el desarrollo de dispositivos Arduino. Destaquen la conexión entre la física y la tecnología.

FROM GAME TO STEM

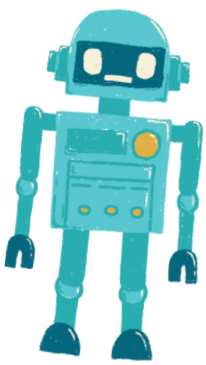
Co-funded by the Erasmus+ Programme of the European Union

Lección 4
Lección de Física con Arduino

Explorando Oscilaciones y Movimiento de Péndulo con Arduino

Duración:

2 períodos



Objetivo:

- Los estudiantes comprenderán los principios del movimiento oscilatorio y las oscilaciones armónicas.
- Los estudiantes aplicarán la modelización matemática para analizar el movimiento del péndulo.
- Los estudiantes programarán un Arduino para simular y visualizar el movimiento del péndulo.

Materiales::

- Placas Arduino (una por estudiante o grupo)
- Servomotores
- Protoboards
- Cables de puente
- Pequeños objetos como pesos de péndulo (por ejemplo, arandelas)
- Cuerda o hilo para péndulos
- Reglas o cinta métrica
- Cables USB para programación
- Computadoras con el entorno de desarrollo Arduino IDE instalado
- Proyector o pizarra blanca para demostraciones

Actividades

Día 1

Introducción a las Oscilaciones y el Movimiento del Péndulo (15 minutos):

- Comienza la lección presentando el concepto de movimiento oscilatorio y su relevancia en diversos campos, incluyendo la física y la ingeniería.
- Define términos clave como período, frecuencia, amplitud y oscilaciones armónicas.
- Explica los fundamentos del movimiento del péndulo y su representación matemática.

Modelado Matemático de Péndulos (30 minutos):

- Discute el modelo matemático de un péndulo simple, incluyendo la ecuación para el período de un péndulo: $T = 2\pi\sqrt{\frac{L}{g}}$
- donde T es el período, L es la longitud del péndulo y g es la aceleración debido a la gravedad.
- Proporciona ejemplos de cómo utilizar la ecuación para calcular el período de un péndulo.
- Realiza cálculos simples relacionados con el movimiento del péndulo.

Introducción a Arduino y los Servomotores (20 minutos):

- Presenta Arduino como una plataforma para crear dispositivos y explica sus componentes (microcontrolador, sensores, actuadores).
- Explica el concepto de servomotores y cómo se pueden utilizar para simular el movimiento del péndulo.
- Muestra ejemplos de control de servomotores con Arduino.

Actividad Práctica (45 minutos):

- Divide a los estudiantes en parejas o grupos pequeños.
- Proporciona a cada grupo una placa Arduino, un servomotor, una protoboard, cables de puente, un pequeño objeto como peso de péndulo y una cuerda.
- Instruye a los estudiantes para que ensamblen un simulador de péndulo simple utilizando el servomotor para controlar el movimiento del péndulo.
- Guía a los estudiantes en la escritura de código de Arduino para crear oscilaciones armónicas variando la posición del servomotor.

Día 2

Revisión del Movimiento del Péndulo (20 minutos):

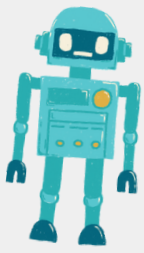
- Comienza el segundo día revisando los conceptos de movimiento oscilatorio, movimiento del péndulo y el modelo matemático de un péndulo simple.
- Discute los proyectos de Arduino del día anterior y sus aplicaciones.

Programación del Simulador de Péndulo (45 minutos):

- Continúa con la actividad práctica del primer día.
- Instruye a los estudiantes para que modifiquen su código de Arduino para simular con precisión el movimiento del péndulo con diferentes parámetros como longitud y amplitud.
- Anima a los estudiantes a visualizar y representar gráficamente el movimiento utilizando el servomotor.

Aquí tienes un ejemplo de código de Arduino para crear un simulador de péndulo simple utilizando un servomotor. Este código permite a los estudiantes programar un Arduino para simular y visualizar el movimiento de un péndulo:

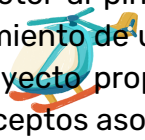
```
#include <Servo.h>
Servo pendulum; // Create a Servo object
int angle = 90; // Initial angle of the pendulum (straight down)
int amplitude = 45; // Maximum angle to one side of the vertical (adjust as needed)
int period = 2000; // Period of the pendulum swing in milliseconds (adjust as needed)
void setup() {
  pendulum.attach(9); // Attach the servo to digital pin 9
}
void loop() {
  // Calculate the angle of the pendulum using harmonic motion
  int displacement = amplitude * cos(2 * PI * millis() / period);
  int pendulumAngle = angle + displacement;
   // Move the servo to the calculated angle
  pendulum.write(pendulumAngle);
  // Delay for a short time to control the speed of the simulation
  delay(50); // Adjust as needed for desired speed
}
```



En este código:

1. Incluimos la biblioteca Servo para controlar el servomotor.
2. Creamos un objeto Servo llamado "pendulum" para controlar el servomotor.
3. Definimos variables para el ángulo inicial del péndulo (angle), el ángulo máximo hacia un lado de la vertical (amplitude) y el período del movimiento del péndulo (period). Puedes ajustar estos valores para cambiar el comportamiento del péndulo.
4. En la función "setup()", adjuntamos el servomotor al pin digital 9.
5. En la función "loop()", calculamos el ángulo del péndulo utilizando la fórmula para el movimiento armónico. El desplazamiento representa el desplazamiento angular desde la posición vertical, y lo sumamos al ángulo inicial para obtener "pendulumAngle".
6. Utilizamos "pendulum.write(pendulumAngle)" para mover el servomotor a un ángulo calculado.
7. Agregamos un retardo para controlar la velocidad de la simulación. Ajusta el valor del retardo según sea necesario para lograr la velocidad de simulación deseada.

Para utilizar este código, los estudiantes deben conectar un servomotor al pin digital 9 del Arduino y cargar el código en la placa. El servomotor simulará el movimiento de un péndulo, y los estudiantes podrán observar las oscilaciones en acción. Este proyecto proporciona una representación visual y práctica del movimiento del péndulo y sus conceptos asociados.



Presentación del Proyecto y Discusión (30 minutos):

- Cada grupo presenta su simulador de péndulo basado en Arduino a la clase.
- Anima a los estudiantes a explicar cómo aplicaron los principios de modelado matemático en sus proyectos.
- Discute las aplicaciones del mundo real de comprender las oscilaciones armónicas en campos como la física, la ingeniería y la astronomía.
- Facilita una discusión en clase sobre los desafíos y soluciones encontrados durante el proyecto.

Evaluaciones

Evalúa a los estudiantes en función de su participación, la calidad de su dispositivo Arduino y su capacidad para explicar los principios del movimiento oscilatorio y las oscilaciones armónicas.

Tarea

Tarea: Investigación y Presentación de Aplicaciones del Mundo Real de las Oscilaciones y el Movimiento Armónico.

Resumen

Concluyan la lección resumiendo los conceptos clave de física aprendidos y enfatizando su uso práctico en el desarrollo de dispositivos Arduino. Destaquen la conexión entre las matemáticas y la tecnología en la comprensión del movimiento del péndulo.

FROM GAME TO STEM

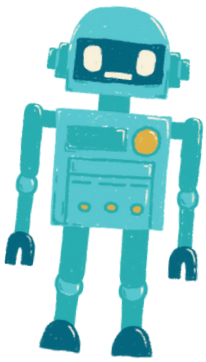
Co-funded by the Erasmus+ Programme of the European Union

Lección 5
Lección de Química con Arduino

Explorando la Hidroponía y la Química del Crecimiento de Plantas con Arduino

Duración:

3 períodos



Objetivo::

- Los estudiantes comprenderán los principios de la hidroponía y sus ventajas en el crecimiento de las plantas.
- Los estudiantes aprenderán sobre los nutrientes esenciales para el crecimiento de las plantas y cómo preparar soluciones nutritivas.
- Los estudiantes diseñarán y construirán un sistema de riego automatizado utilizando Arduino.
- Los estudiantes supervisarán y controlarán los niveles de pH en un sistema hidropónico para optimizar el crecimiento de las plantas.

Materiales:

- Placas Arduino (una por estudiante o grupo)
- Sensores de pH y soluciones de calibración de pH
- Bombas de agua
- Tubos y emisores de goteo
- Reservorio para la solución de nutrientes
- Soluciones de pH alto y pH bajo
- Soluciones de buffer de pH
- Semillas o plántulas de plantas
- Medio de cultivo hidropónico (por ejemplo, lana de roca, perlita)
- Componentes de la solución de nutrientes (por ejemplo, fertilizante N-P-K)
- Recipientes para configuraciones hidropónicas
- Cables de puente
- Cables USB para programación
- Computadoras con el entorno de desarrollo Arduino IDE instalado
- Proyector o pizarra blanca para demostraciones

Actividades

Día1

Introducción a la Hidroponía (15 minutos):

- Comienza la lección presentando el concepto de hidroponía y sus ventajas en el crecimiento de las plantas.
- Discute los beneficios de los ambientes controlados y los sistemas eficientes en el uso del agua.

Química del Crecimiento de las Plantas (30 minutos):

- Explica los elementos químicos esenciales para el crecimiento de las plantas (por ejemplo, nitrógeno, fósforo, potasio) y sus funciones.
- Habla sobre la importancia de los niveles de pH en la absorción de nutrientes y la salud de las plantas.
- Introduce el concepto de soluciones nutritivas y el control de pH en la hidroponía.

Introducción a Arduino y Sensores (20 minutos):

- Presenta Arduino como una plataforma para la automatización y la recopilación de datos, y explica sus componentes (microcontrolador, sensores).
- Muestra ejemplos de sensores utilizados en sistemas hidropónicos (por ejemplo, sensores de pH).

Preparación de la Actividad Práctica (45 minutos):

- Proporciona a los estudiantes placas Arduino, sensores de pH, bombas de agua, tubos y recipientes.
- Instruye a los estudiantes a que hagan una lluvia de ideas y planifiquen el diseño de su sistema hidropónico, incluyendo las soluciones nutritivas y el control de pH.

Día2

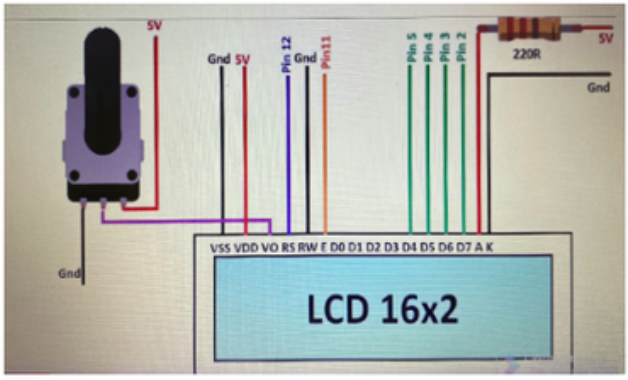
Configuración del Sistema Hidropónico (60 minutos):

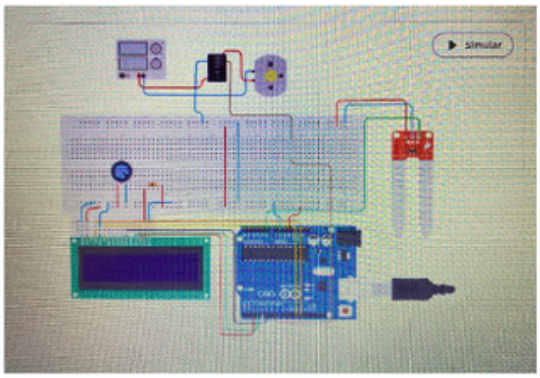
- Comienza el segundo día permitiendo que los estudiantes configuren sus sistemas hidropónicos.
- Los estudiantes deben plantar semillas o plántulas en un medio de cultivo hidropónico (por ejemplo, lana de roca) y organizar el sistema de riego con tubos y emisores de goteo.
- Demuestra cómo mezclar y preparar las soluciones nutritivas.

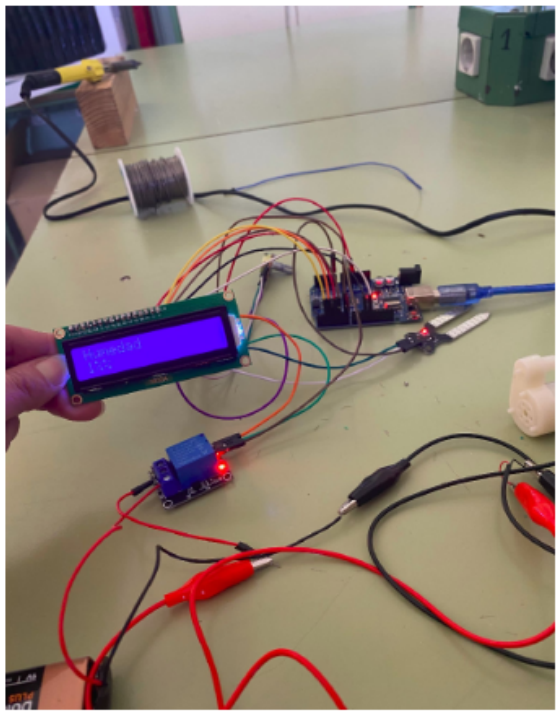
Monitoreo y Calibración de pH (30 minutos):

- Instruye a los estudiantes sobre cómo calibrar y utilizar los sensores de pH para el monitoreo de pH.
- Explica la importancia de mantener el pH dentro del rango óptimo para la absorción de nutrientes (generalmente alrededor de pH 6-7).
- Guía a los estudiantes en la calibración de sus sensores de pH utilizando soluciones buffer de pH.

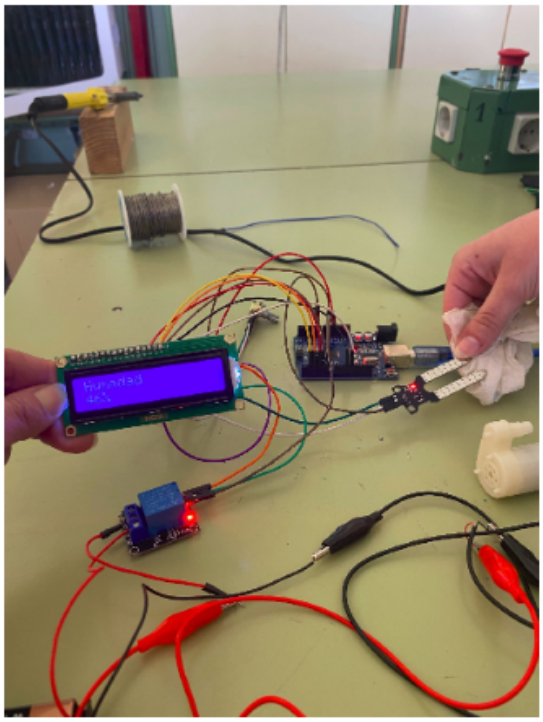
Diagrama de las Conexiones:







Cuando la humedad está al 1%, la bomba funciona.



Cuando la humedad está al 46%, la bomba sigue funcionando.

Día3

Programación de Arduino (60 minutos):

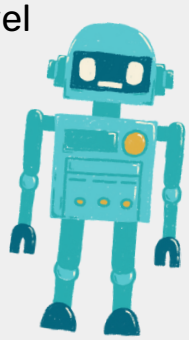
- Instruye a los estudiantes para que escriban código de Arduino para el sistema de rieg automatizado.
- Los estudiantes deben programar el Arduino para controlar el horario de la bomba de agua y monitorear los niveles de pH.
- Enfatiza la importancia del riego regular y los ajustes de pH para el crecimiento de las plantas.

Recopilación de Datos y Control de pH (30 minutos):

- Explica cómo recopilar datos de los sensores de pH y monitorear los niveles de pH.
- Discute las acciones que deben tomar los estudiantes si los niveles de pH se desvían del rango óptimo.

Aquí tienes un ejemplo de código de Arduino para un sistema de riego hidropónico automatizado con monitoreo y control de pH. Este código es un marco básico que puedes adaptar y ampliar según tu configuración y necesidades específicas.

```
#include <Adafruit_ADS1015.h>
#include <Wire.h>
#define PH_SENSOR_PIN 0 // Analog pin for pH sensor
#define PUMP_PIN 12    // Digital pin for water pump
// Create an Adafruit ADS1015 ADC object
Adafruit_ADS1015 ads;
float currentpH;
float targetpH = 6.5; // Adjust this value to your desired pH level
void setup() {
 Serial.begin(9600);
 ads.begin();
 pinMode(PUMP_PIN, OUTPUT);
 digitalWrite(PUMP_PIN, LOW); // Initialize pump as off
}
void loop() {
 // Read pH value from pH sensor
 currentpH = readpH();
 // Display current pH value
 Serial.print("Current pH: ");
 Serial.println(currentpH);
 // Check and adjust pH level
 if (currentpH < targetpH) {
  // pH is too low, add pH up solution (adjust as needed)
  // Implement pH adjustment mechanism here
    // For example, you can control a peristaltic pump for adding pH up solution
```



```
digitalWrite(PUMP_PIN, HIGH); // Turn on the pump
 delay(1000); // Adjust the delay time as needed
 digitalWrite(PUMP_PIN, LOW); // Turn off the pump
 } else if (currentpH > targetpH) {
 // pH is too high, add pH down solution (adjust as needed)
 // Implement pH adjustment mechanism here
  // For example, you can control a peristaltic pump for adding pH down solution
 digitalWrite(PUMP_PIN, HIGH); // Turn on the pump
 delay(1000); // Adjust the delay time as needed
 digitalWrite(PUMP_PIN, LOW); // Turn off the pump
 }
 // Add code for irrigation control here (e.g., based on time intervals)
}
float readpH() {
 // Read pH value from the pH sensor and convert to pH scale
 int16_t rawValue = ads.readADC_SingleEnded(PH_SENSOR_PIN);
 float voltage = (rawValue * 0.1875) / 1000.0; // Convert to voltage
  float pHValue = 3.5 * voltage + 3.5; // Convert to pH scale (adjust calibration values as needed)
 return pHValue;
}
```

En este código:

- Utilizamos la biblioteca Adafruit ADS1015 para comunicarnos con el convertidor analógico-digital ADS1015, que se utiliza para leer los valores del sensor de pH. Asegúrate de tener esta biblioteca instalada en tu entorno de desarrollo de Arduino.
- La función setup() inicializa la comunicación serial para la salida de datos, inicializa el ADC y configura el pin de la bomba de agua como salida.
- En la función loop(), leemos continuamente el valor de pH del sensor de pH utilizando la función readpH().
- Comparamos el valor de pH actual con el nivel de pH objetivo (targetpH) y ajustamos el pH según sea necesario. Debes implementar el mecanismo de ajuste de pH según tu configuración específica, que puede implicar el control de una bomba peristáltica para agregar soluciones de pH alto o pH bajo.
- Además, puedes agregar código para controlar la bomba de agua para el riego según intervalos de tiempo u otros criterios.

Ten en cuenta que este código proporciona un marco básico y es posible que debas calibrarlo y ajustarlo según tu configuración específica de sensor y bomba, así como los niveles de pH deseados y el programa de riego. Asegúrate de tomar precauciones de seguridad al trabajar con productos químicos y bombas.

Día4

Configuración del Experimento y Monitoreo (60 minutos):

- Permite que los estudiantes configuren sus sistemas hidropónicos automatizados en el invernadero o el aula.
- Los estudiantes deben poner en marcha el sistema y supervisar el crecimiento de las plantas, los niveles de nutrientes y el pH.

Presentación del Proyecto y Discusión (60 minutos):

- Cada grupo presenta su diseño de sistema hidropónico, metodología y resultados iniciales a la clase.
- Discute el impacto de las soluciones nutritivas y el control de pH en el crecimiento de las plantas.
- Anima a los estudiantes a proponer optimizaciones para sus sistemas basadas en observaciones iniciales.

Evaluaciones

Evalúa a los estudiantes en función de su participación, configuración del sistema, calibración de pH, recopilación de datos, análisis y la calidad de sus presentaciones.

Tarea

Asigna la siguiente Tarea que requiere que los estudiantes investiguen y presenten una aplicación del mundo real de la hidroponía en la agricultura o la agricultura sostenible.

Conclusión

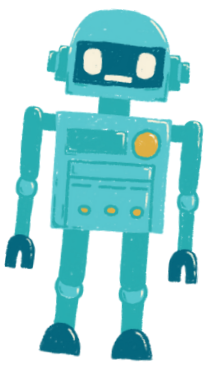
Concluimos esta lección de química reforzando los conceptos clave aprendidos y destacando la importancia de la hidroponía en la agricultura sostenible y la producción de alimentos. También resaltamos el papel fundamental de la tecnología, en particular Arduino, en la automatización y optimización de los sistemas hidropónicos.

Lección 6
Lección de Química con Arduino

Explorando Reacciones Químicas con Arduino

Duración:

3 períodos



Objetivo:

- Los estudiantes comprenderán los principios de las reacciones químicas y su relevancia en la vida cotidiana.
- Los estudiantes diseñarán y llevarán a cabo experimentos para observar y medir reacciones químicas.
- Los estudiantes programarán un sistema de monitoreo de temperatura basado en Arduino para recopilar y analizar datos de reacciones químicas.

Materiales:

- Placas Arduino (una por estudiante o grupo)
- Sensores de temperatura (por ejemplo, DS18B20 o LM35)
- Productos químicos para experimentos (por ejemplo, bicarbonato de sodio, vinagre)
- Recipientes para reacciones (por ejemplo, vasos de precipitados, tubos de ensayo)
- Cables de puente
- Cables USB para programación
- Computadoras con el entorno de desarrollo Arduino IDE instalado
- Proyector o pizarra blanca para demostraciones
- Gafas de seguridad y batas de laboratorio

Actividades

Día1

Introducción a las Reacciones Químicas (15 minutos):

- Comienza la lección presentando el concepto de reacciones químicas y su papel en la vida diaria.
- Discute la importancia de las reacciones químicas en diversos campos, incluyendo la química, la biología y la industria.

Conceptos Básicos de las Reacciones Químicas (30 minutos):

- Profundiza en los fundamentos de las reacciones químicas, incluyendo reactantes, productos y la conservación de la masa.
- Proporciona ejemplos de reacciones químicas comunes y sus aplicaciones.
- Enfatiza los protocolos de seguridad al manipular productos químicos en experimentos.

Introducción a Arduino y Sensores (20 minutos):

- Presenta Arduino como una plataforma para la recopilación de datos y explica sus componentes (microcontrolador, sensores).
- Muestra ejemplos de sensores de temperatura y cómo se pueden conectar a Arduino para la recopilación de datos.

Preparación de la Actividad Práctica (45 minutos):

- Proporciona a los estudiantes placas Arduino, sensores de temperatura, productos químicos y recipientes para reacciones.
- Instruye a los estudiantes a que hagan una lluvia de ideas y planifiquen sus experimentos de reacciones químicas, centrándose en los cambios de temperatura.
- Discute las precauciones de seguridad y las normas de laboratorio.

Día2

Configuración del experimento y recopilación de datos (60 minutos):

- Comience el segundo Día permitiendo a los estudiantes configurar sus experimento de reacciones químicas.
- Los estudiantes deben mezclar los químicos seleccionados en recipientes de reacción y colocar los sensores de temperatura.
- Indique a los estudiantes que escriban código Arduino para monitorear la temperatura y recopilar datos.

Análisis y discusión de datos (30 minutos):

- Guíe a los estudiantes en el análisis de sus datos, buscando cambios de temperatura durante las reacciones químicas.
- Discuta la importancia de los cambios de temperatura en las reacciones químicas y los conceptos de reacciones exotérmicas y endotérmicas.
- Anime a los estudiantes a sacar conclusiones basadas en sus hallazgos.

Día3

Programación del Sistema de Monitoreo (60 minutos):

- Instruya a los estudiantes a ajustar su código Arduino para la recopilación y el análisis de datos.
- Los estudiantes deben programar el Arduino para registrar datos de temperatura a intervalos específicos y mostrarlos gráficamente (por ejemplo, en una pantalla LCD o en el monitor serie).

Presentación y discusión del proyecto (60 minutos):

- Cada grupo presenta a la clase la configuración, la metodología y los resultados de su experimento de reacción química.
- Anime a los estudiantes a explicar sus observaciones y las implicaciones de los cambios de temperatura en las reacciones químicas.
- Facilite una discusión en clase sobre las aplicaciones del mundo real de reacciones químicas en diversos campos.

A continuación se muestra un ejemplo de código Arduino que puede utilizar para controlar la temperatura durante un experimento de reacción química. Este código lee datos de temperatura de un sensor de temperatura DS18B20 y los muestra en el monitor serie. Puede personalizar y ampliar este código para adaptarlo a su experimento específico.

```
#include <OneWire.h>
#include <DallasTemperature.h>
// Data wire is plugged into digital pin 2
#define ONE_WIRE_BUS 2
// Create a OneWire object
OneWire oneWire(ONE_WIRE_BUS);
// Pass the OneWire reference to Dallas Temperature sensor
DallasTemperature sensors(&oneWire);
```

```
void setup() {
 // Initialize serial communication for data output
 Serial.begin(9600);

 // Start the temperature sensor library
 sensors.begin();
}

void loop() {
 // Request temperature readings
 sensors.requestTemperatures();

 // Read temperature in Celsius
 float temperatureC = sensors.getTempCByIndex(0);

 // Display temperature on the Serial Monitor
 Serial.print("Temperature: ");
 Serial.print(temperatureC);
 Serial.println(" °C");

 // Add code here for data storage or further analysis

 // Delay before the next temperature reading (adjust as needed)
 delay(1000); // 1-second delay
}
```

En este código:

Usamos la biblioteca Dallas Temperature para interactuar con el sensor de temperatura DS18B20. Asegúrese de tener esta biblioteca instalada en su IDE de Arduino.

La función setup() inicializa la comunicación serial para la salida de datos e inicia la biblioteca de sensores de temperatura.
En la función loop(), el programa solicita y lee continuamente datos de temperatura del sensor usando sensors.getTempCByIndex(0). El 0 indica el primer (y en este caso, el único) sensor de temperatura conectado.
Los datos de temperatura se muestran en el monitor serie con un retraso de 1 segundo entre lecturas. Puede ajustar el tiempo de retraso para controlar la frecuencia de recopilación de datos.

Puede modificar este código para incluir sensores adicionales para diferentes experimentos, implementar el almacenamiento de datos en una tarjeta SD o crear representaciones gráficas de los datos en una pantalla LCD, según sus requisitos específicos.

Evaluaciones

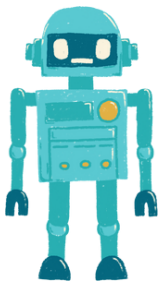
Evaluar a los estudiantes en función de su participación en el experimento, la recopilación de datos, el análisis y la calidad de sus presentaciones.

Tarea

Asigne Tarea que requiera que los estudiantes investiguen y presenten una aplicación del mundo real de reacciones químicas en una industria o campo de la ciencia específico.

Conclusión

Concluya la lección resumiendo los conceptos clave de química aprendidos y reforzando la importancia de las reacciones químicas para comprender los procesos naturales y los avances tecnológicos. Destacar el papel de la tecnología (Arduino) en las investigaciones científicas.



FROM GAME TO STEM

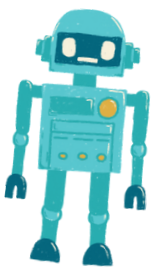
Co-funded by the Erasmus+ Programme of the European Union

Lección 7
Lección de Biología con Arduino

Investigación de la Conductividad Térmica con Termómetros Arduino

Duración:

3 períodos



Objetivo:

- Los estudiantes comprenderán el concepto de conductividad térmica y su importancia.
- Los estudiantes diseñarán y llevarán a cabo un experimento para determinar la conductividad térmica de diferentes materiales.
- Los estudiantes utilizarán termómetros Arduino para recopilar datos de temperatura y analizar sus hallazgos.

Materiales:

- Placas Arduino (una por estudiante o grupo)
- Sensores de temperatura (por ejemplo, DS18B20 o LM35)
- Placas de pruebas (breadboards)
- Cables de puente (jumper wires)
- Diversos materiales para probar la conductividad (por ejemplo, metales, plásticos, madera, vidrio)
- Materiales aislantes (por ejemplo, espuma, tela)
- Agua caliente o una fuente de calor
- Agua fría o hielo
- Cronómetro o temporizador
- Cables USB para programación
- Computadoras con el entorno de desarrollo Arduino IDE instalado
- Proyector o pizarra blanca para demostraciones
- Gafas de seguridad y guantes (para manipular materiales calientes)

Actividades

Día 1

Introducción a la Conductividad Térmica (15 minutos):

- Comienza la lección presentando el concepto de conductividad térmica y por qué es importante en la vida cotidiana.
- Discute aplicaciones del mundo real de la conductividad térmica, como en la cocina, los materiales de construcción y la ingeniería.

Transferencia de Calor y Aislamiento (30 minutos):

- Explica los diferentes métodos de transferencia de calor (conducción, convección, radiación) y concéntrate en la conducción, que es relevante para el experimento.
- Habla sobre los materiales aislantes y su papel en la reducción de la transferencia de calor.
- Enfatiza la necesidad de un experimento controlado para medir la conductividad térmica.

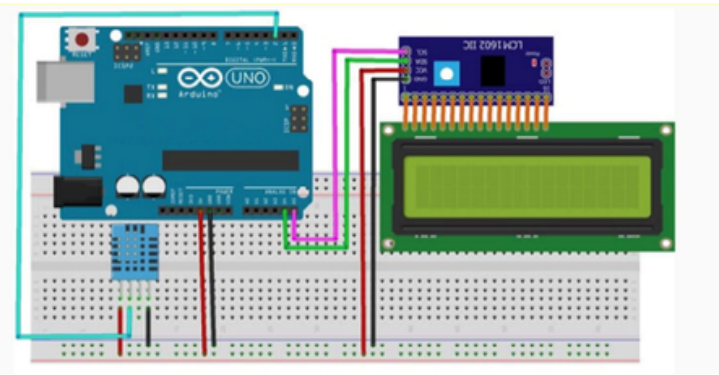
Introducción a los Termómetros Arduino (20 minutos):

- Presenta Arduino como una plataforma para la recopilación de datos y explica sus componentes (microcontrolador, sensores).
- Explica el uso de los sensores de temperatura y cómo se pueden conectar a las placas Arduino.
- Muestra ejemplos de recopilación y visualización de datos de temperatura con Arduino.

Preparación de la Actividad Práctica (45 minutos):

- Proporciona a los estudiantes placas Arduino, sensores de temperatura, placas de pruebas y cables de puente.
- Explica la configuración experimental: cada grupo probará diferentes materiales para determinar su conductividad térmica.
- Instruye a los estudiantes a que hagan una lluvia de ideas y planifiquen sus experimentos, incluyendo cómo controlar variables y recopilar datos.

ESQUEMA:



Día 2

Configuración del Experimento y Recopilación de Datos (60 minutos):

- Comienza el segundo día permitiendo que los estudiantes configuren sus experimentos.
- Los estudiantes deben conectar el sensor de temperatura a los materiales seleccionados y preparar contenedores de agua caliente y fría.
- Instruye a los estudiantes a que comiencen a recopilar datos de temperatura utilizando Arduino y registren las temperaturas iniciales.
- Pide a los estudiantes que midan y registren el tiempo que tarda en cambiar la temperatura en cada material.

Análisis de Datos y Discusión (30 minutos):

- Guía a los estudiantes en el análisis de sus datos, el cálculo de la tasa de cambio de temperatura y la comparación de la conductividad de diferentes materiales.
- Discute el concepto de resistencia térmica y cómo se relaciona con la conductividad.
- Anima a los estudiantes a identificar cuál es el mejor conductor de calor y cuál es el peor basándose en sus hallazgos.

Aquí tienes un ejemplo de código Arduino que puedes usar para el experimento de medición y comparación de la conductividad térmica de diferentes materiales utilizando un sensor de temperatura (por ejemplo, DS18B20). Este código recopilará datos de temperatura del sensor y te permitirá calcular la tasa de cambio de temperatura para cada material.

```
#include <OneWire.h>
#include <DallasTemperature.h>
// Data wire is plugged into digital pin 2
#define ONE_WIRE_BUS 2
// Create a OneWire object
OneWire oneWire(ONE_WIRE_BUS);
// Pass the OneWire reference to Dallas Temperature sensor
DallasTemperature sensors(&oneWire);
void setup() {
 // Initialize serial communication for data output
 Serial.begin(9600);
 // Start the temperature sensor library
 sensors.begin();
}
void loop() {
```

```
// Initialize variables for temperature measurements
 float initialTemp, finalTemp;
 unsigned long startTime, endTime;
 float rateOfChange;
 // Wait for the user to start the experiment (e.g., press a button)
while (!digitalRead(3)) {
  // Wait for button press
 }
 delay(1000); // Debounce delay
 // Measure initial temperature
 sensors.requestTemperatures(); // Request temperature readings
 initialTemp = sensors.getTempCByIndex(0); // Get temperature in Celsius
 // Record the start time
 startTime = millis();
 // Wait for the temperature to stabilize (e.g., 5 seconds)
 delay(5000);
 // Measure final temperature
 sensors.requestTemperatures();
 finalTemp = sensors.getTempCByIndex(0);
 // Record the end time
 endTime = millis();
 // Calculate the rate of temperature change (°C per second)
  rateOfChange = (finalTemp - initialTemp) / ((endTime - startTime) /
1000.0);
 // Output results to the Serial Monitor
 Serial.print("Initial Temperature: ");
 Serial.print(initialTemp);
 Serial.println(" °C");
 Serial.print("Final Temperature: ");
 Serial.print(finalTemp);
 Serial.println(" °C");
 Serial.print("Rate of Change: ");
 Serial.print(rateOfChange);
 Serial.println(" °C/s");
 // Wait for user input (e.g., press a button) to proceed to the next material
 Serial.println("Press a button to test the next material.");
 while (!digitalRead(3)) {
  // Wait for button press
 }
 delay(1000); // Debounce delay
}
```

En este código:
Utilizamos la biblioteca Dallas Temperature para interactuar con el sensor de temperatura DS18B20. Asegúrate de tener esta biblioteca instalada en tu entorno de desarrollo de Arduino.
Definimos el pin digital (por ejemplo, pin 2) al que está conectado el cable de datos del sensor DS18B20.
En la función setup(), inicializamos la comunicación serial para la salida de datos y comenzamos la biblioteca del sensor de temperatura.
En la función loop(), el programa espera a que se presione un botón para iniciar el experimento. Se mide la temperatura antes y después de un período de espera (por ejemplo, 5 segundos) para calcular la tasa de cambio de temperatura.
La tasa de cambio se calcula como la diferencia de temperatura dividida por el tiempo necesario para alcanzar ese cambio. Esto te proporciona la tasa en °C por segundo.
Los resultados se imprimen en el Monitor Serie, incluyendo la temperatura inicial, la temperatura final y la tasa de cambio.
Después de que se complete el experimento para un material, puedes presionar un botón para proceder al siguiente material.
Recuerda conectar correctamente el sensor DS18B20 al Arduino y asegurarte de que el botón esté conectado a un pin digital (por ejemplo, pin 3) para activar el experimento. Ajusta el código según sea necesario en función de tu configuración específica.

Día 3

Presentación del Proyecto y Discusión (60 minutos):

- Cada grupo presenta su configuración experimental, metodología y resultados a la clase.
- Anima a los estudiantes a explicar sus observaciones, discutir posibles fuentes de error y sugerir mejoras.
- Facilita una discusión en clase sobre las implicaciones del mundo real de sus hallazgos, como la selección de materiales para el aislamiento térmico o materiales conductores para disipadores de calor.

Evaluaciones

Evalúa a los estudiantes en función de su participación en el experimento, la recopilación de datos, el análisis y la calidad de sus presentaciones.

Tarea

Asigna una tarea que requiera a los estudiantes investigar y presentar una aplicación del mundo real de la conductividad térmica en la ingeniería o la ciencia de materiales.

Conclusión

Concluye la lección resumiendo los conceptos clave aprendidos y reforzando la importancia de comprender la conductividad térmica en la vida cotidiana y las aplicaciones tecnológicas.

FROM GAME TO STEM

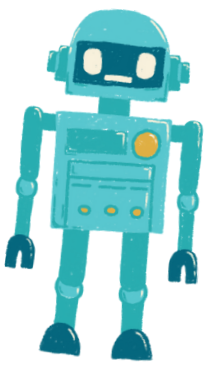
Co-funded by the Erasmus+ Programme of the European Union

Lección 8
Lección de Biología con Arduino

Explorando la Fotosíntesis y el Crecimiento de las Plantas con Arduino

Duración:

3 períodos



Objetivo:

- Los estudiantes comprenderán el proceso de la fotosíntesis y su importancia en el crecimiento de las plantas.
- Los estudiantes diseñarán y llevarán a cabo un experimento para investigar el efecto de los factores ambientales en la fotosíntesis.
- Los estudiantes programarán un sistema basado en Arduino para monitorear y recopilar datos relacionados con el crecimiento de las plantas.

Materiales:

- Placas Arduino (una por estudiante o grupo)
- Sensores (por ejemplo, sensor de luz, sensor de temperatura, sensor de humedad)
- Luces LED de crecimiento (opcional)
- Plantas en macetas o semillas
- Tierra y contenedores para plantar
- Cables de puente
- Cables USB para programación
- Computadoras con Arduino IDE instalado
- Proyector o pizarra blanca para demostraciones
- Tierra para macetas, agua y otros materiales para el cuidado de las plantas

Actividades

Día 1

<u>Introducción a la Fotosíntesis (15 minutos):</u>

- Comienza la lección presentando el concepto de fotosíntesis y su importancia en el crecimiento de las plantas.
- Discute la ecuación química de la fotosíntesis y el papel de la luz, el dióxido de carbono y el agua en el proceso.

<u>Factores Ambientales y Crecimiento de las Plantas (30 minutos):</u>

- Explica cómo varios factores ambientales, como la intensidad lumínica, la temperatura y la humedad, pueden afectar la fotosíntesis y el crecimiento de las plantas.
- Habla sobre por qué estos factores son esenciales para el desarrollo saludable de las plantas.
- Enfatiza la necesidad de realizar experimentos controlados para estudiar estos factores.

<u>Introducción a Arduino y Sensores (20 minutos):</u>

- Presenta Arduino como una plataforma para la recopilación de datos y explica sus componentes (microcontrolador, sensores).
- Muestra ejemplos de sensores comúnmente utilizados en la monitorización ambiental y el cuidado de las plantas.
- Explica el papel de los sensores en la recopilación de datos para experimentos.

<u>Preparación de la Actividad Práctica (45 minutos):</u>

- Proporciona a los estudiantes placas Arduino, sensores (por ejemplo, sensor de luz, sensor de temperatura, sensor de humedad) y luces LED de crecimiento (opcional).
- Instruye a los estudiantes a que realicen lluvias de ideas y planifiquen sus experimentos de crecimiento de plantas, centrándose en un factor ambiental.
- Pide a los estudiantes que preparen macetas con tierra y planten semillas o pequeñas plantas en macetas.

Día 2

Configuración del Experimento y Recopilación de Datos (60 minutos):

- Comienza el segundo día permitiendo que los estudiantes configuren sus experimentos.
- Los estudiantes deben colocar los sensores en el entorno de las plantas, conectarlos a Arduino y posicionar las luces LED de crecimiento (si las están utilizando).
- Instruye a los estudiantes a recopilar datos relacionados con el factor ambiental elegido, como la intensidad lumínica o la temperatura.
- Enfatiza la importancia de registrar datos de manera precisa y regular.

Análisis de Datos y Discusión (30 minutos):

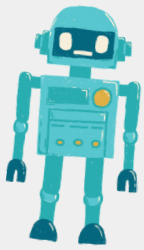
- Guía a los estudiantes en el análisis de sus datos, buscando tendencias o patrones relacionados con el crecimiento de las plantas y el factor elegido.
- Discute el impacto del factor en la fotosíntesis y el desarrollo de las plantas.
- Anima a los estudiantes a sacar conclusiones a partir de sus hallazgos.

Día 3

Programming the Monitoring System (60 minutes):

- Instruct students to write Arduino code to monitor and collect data from the sensors.
- Students should program the Arduino to record data at specific intervals (e.g., every hour).
- Discuss how to use libraries for sensor data retrieval and how to store data.

Aquí tienes un ejemplo de código Arduino para un sistema simple de monitorización ambiental que mide y registra datos relacionados con la intensidad de la luz utilizando un sensor de luz. Puedes adaptar este código para otros factores ambientales según sea necesario:



```
#include <Wire.h>
#include <Adafruit_Sensor.h>
#include <Adafruit_TSL2591.h>
Adafruit_TSL2591 tsl = Adafruit_TSL2591(2591);
void setup() {
 Serial.begin(9600);
  // Initialize the light sensor
 if(!tsl.begin()) {
   Serial.println("Light sensor not found. Check wiring.");
   while(1);
 }
 tsl.setGain(TSL2591_GAIN_LOW);    // Adjust the gain (options: LOW, MED, HIGH, MAX)
 tsl.setTiming(TSL2591_INTEGRATIONTIME_100MS); // Adjust the integration time (options: 100, 200, 300, 400, 500, 600)
}
void loop() {
 // Read and print light intensity data
 uint16_t luminance = tsl.getLuminosity(TSL2591_VISIBLE);
 Serial.print("Light Intensity (Lux): ");
 Serial.println(luminance);
    // Add code here to log data to an SD card, display on an LCD, or transmit to a computer/server.
 // Wait for a specific time interval (e.g., 1 hour)
 delay(3600000); // Adjust the delay time as needed
}
```

En este código:

Utilizamos la biblioteca Adafruit TSL2591 para interactuar con el sensor de luz TSL2591. Asegúrate de tener esta biblioteca instalada en tu entorno de desarrollo de Arduino.
La función setup() inicializa la comunicación serial para la salida de datos y configura el sensor de luz. También configura la ganancia y el tiempo de integración del sensor según tus necesidades.
En la función loop(), el programa lee continuamente los datos de intensidad de luz (en lux) del sensor utilizando tsl.getLuminosity(TSL2591_VISIBLE).
Puedes agregar código dentro del bucle para registrar los datos en una tarjeta SD, mostrarlos en una pantalla LCD o transmitirlos a una computadora o servidor para un análisis posterior. Por ejemplo, puedes utilizar un módulo de tarjeta SD para almacenar los datos localmente.
El retardo al final del bucle está configurado para esperar un intervalo de tiempo específico (por ejemplo, 1 hora) antes de tomar la siguiente lectura. Puedes ajustar el tiempo de retardo para controlar la frecuencia de recopilación de datos.

Este código proporciona un marco básico para monitorear la intensidad de luz, y puedes ampliarlo agregando más sensores para monitorear factores ambientales adicionales como la temperatura y la humedad. Recuerda ajustar el código para que coincida con los sensores y el hardware específicos que estás utilizando en tu experimento.

Presentación del Proyecto y Discusión (60 minutos):

- Cada grupo presenta su experimento de crecimiento de plantas, la configuración, la metodología y los resultados a la clase.
- Anima a los estudiantes a explicar sus observaciones y las implicaciones de sus hallazgos para el cuidado de las plantas y la agricultura.
- Facilita una discusión en clase sobre la importancia de la fotosíntesis en el ecosistema y cómo la tecnología (Arduino) puede ayudar en las investigaciones científicas.

Evaluaciones

Evalúa a los estudiantes en función de su participación en el experimento, la recopilación de datos, el análisis y la calidad de sus presentaciones.

Tarea

Asigna una tarea que requiera a los estudiantes investigar y presentar una aplicación del mundo real de la fotosíntesis y la monitorización ambiental en la agricultura o la ecología.

Conclusión

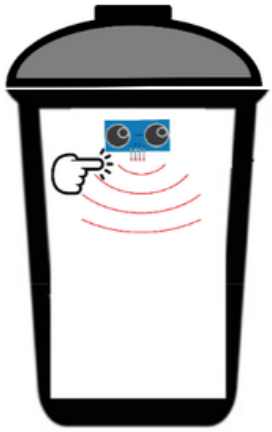
En esta lección, hemos explorado el fascinante mundo de la fotosíntesis y su importancia en el crecimiento de las plantas. Los estudiantes han tenido la oportunidad de diseñar y llevar a cabo experimentos para investigar cómo factores ambientales como la luz, la temperatura y la humedad afectan la fotosíntesis y, en última instancia, la salud de las plantas.

FROM GAME TO STEM

Co-funded by the Erasmus+ Programme of the European Union

Lección 9
Lección de Ecología de Arduino

Construyendo un Smart Dustbin con Arduino



Duración:

3 períodos

Objetivo::

- Los estudiantes comprenderán los conceptos básicos de la programación de Arduino y sus aplicaciones en la automatización.
- Los estudiantes aprenderán sobre los sensores ultrasónicos y cómo se pueden utilizar para la detección de objetos.
- Los estudiantes construirán un prototipo de Smart Dustbin y lo programarán para que abra su tapa cuando se detecte un objeto.
- Los estudiantes explorarán los beneficios ambientales de los sistemas inteligentes de gestión de residuos.

Materiales::

- Placas Arduino Uno (una por estudiante o grupo)
- Sensor ultrasónico HC-SR04 (uno por estudiante o grupo)
- Servomotores (uno por estudiante o grupo)
- Tableros de conexiones y cables puente
- Pequeña caja de cartón o contenedor (para el basurero)
- Tapa de cartón o plástico (para simular la tapa del basurero)
- Cables USB para la programación
- Computadoras con el IDE de Arduino instalado
- Proyector o pizarra para las demostraciones

Actividades:

Día1

Introducción a Arduino y Electrónica (15 minutos):

- Comienza la lección presentando Arduino como una plataforma de microcontrolador utilizada para diversos proyectos.
- Discute la importancia de la electrónica y la automatización en la tecnología moderna.

Sensores Ultrasónicos y Detección de Objetos (30 minutos):

- Explica el principio de los sensores ultrasónicos y cómo funcionan para la medición de distancias.
- Habla sobre los componentes del sensor ultrasónico HC-SR04 (transmisor y receptor).
- Ilustra cómo se pueden utilizar los sensores ultrasónicos para detectar objetos y medir distancias.

Introducción a los Servomotores (20 minutos):

- Presenta los servomotores y sus aplicaciones en el control de movimientos mecánicos.
- Muestra ejemplos de cómo se pueden utilizar los servomotores en proyectos, como abrir una tapa.

Preparación de la Actividad Práctica (45 minutos):

- Proporciona a cada grupo un Arduino Uno, un sensor ultrasónico, un servomotor, una placa de conexiones y cables puente.
- Instruye a los estudiantes para que ensamblen el prototipo del Smart Dustbin, posicionando adecuadamente el sensor ultrasónico y el servomotor.

Día2

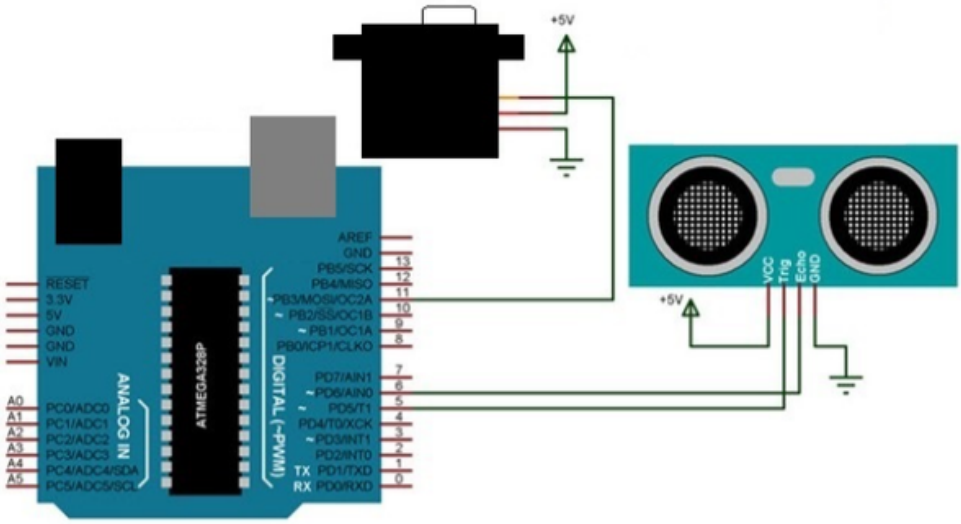
Fundamentos de la Programación en Arduino (30 minutos):

- Enseña a los estudiantes los fundamentos de la programación en Arduino, incluyendo setup(), loop(), y pinMode().
- Proporciona ejemplos de código básico de Arduino para hacer parpadear un LED.

Programación del Smart Dustbin (60 minutos):

- Guía a los estudiantes en la escritura de código Arduino para controlar el Smart Dustbin basado en las lecturas del sensor ultrasónico.
- Explica la lógica para abrir la tapa cuando se detecta un objeto dentro de un rango específico.

shemá:



Aquí tienes un ejemplo de código Arduino para un Smart Dustbin que utiliza un sensor ultrasónico (HC-SR04) y un servomotor. Este código permite que el servomotor abra la tapa del basurero cuando se detecta un objeto dentro de un rango específico:

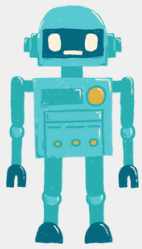
```
#include <Servo.h>

#define TRIGGER_PIN 9
#define ECHO_PIN 10
#define SERVO_PIN 11

Servo myservo; // Create a Servo object
int distance; // Variable to store distance measured by the Ultrasonic sensor

void setup() {
 myservo.attach(SERVO_PIN); // Attach the Servo to the specified pin
 pinMode(TRIGGER_PIN, OUTPUT);
 pinMode(ECHO_PIN, INPUT);
 Serial.begin(9600); // Initialize serial communication for debugging
}
```

```
void loop() {
 // Send a brief pulse to trigger the Ultrasonic sensor
 digitalWrite(TRIGGER_PIN, LOW);
 delayMicroseconds(2);
 digitalWrite(TRIGGER_PIN, HIGH);
 delayMicroseconds(10);
 digitalWrite(TRIGGER_PIN, LOW);
 // Read the duration of the echo pulse and calculate the distance
 duration = pulseIn(ECHO_PIN, HIGH);
 distance = (duration / 2) / 29.1; // Calculate distance in centimeters
 // Check if an object is within the specified range (adjust as needed)
 if (distance < 20) { // You can adjust the distance threshold here
  // If an object is detected, open the flap
   myservo.write(90); // Rotate the Servo to open the flap (adjust the angle as needed)
  delay(1000); // Wait for 1 second
  myservo.write(0); // Rotate the Servo back to close the flap
 }
 // Print the distance to the Serial Monitor for debugging
 Serial.print("Distance: ");
 Serial.print(distance);
 Serial.println(" cm");
 // Add a delay between readings to prevent rapid triggering
 delay(1000); // You can adjust the delay time as needed
}
```



En este código:

- Incluimos la biblioteca Servo y definimos los números de pin para el trigger y echo del sensor ultrasónico, así como el pin de control del servomotor.
- En la función setup(), adjuntamos el servomotor al pin especificado y configuramos el pin trigger como salida y el pin echo como entrada. También inicializamos la comunicación serial para fines de depuración.
- La función loop() realiza repetidamente los siguientes pasos:
  1. Envía un breve pulso al sensor ultrasónico para activar una medición de distancia.
  2. Mide la duración del pulso de eco y calcula la distancia en centímetros.
  3. Comprueba si un objeto está dentro del rango especificado (20 cm en este ejemplo) y, si es así, abre la tapa del basurero girando el servomotor a un ángulo especificado (90 grados).
  4. Imprime la distancia en el Monitor Serial con fines de depuración.
  5. Agrega un retraso entre las lecturas para evitar activaciones rápidas.

Puedes ajustar el umbral de distancia, el ángulo del servomotor y los tiempos de retraso para que se adapten a tu configuración específica y tus requisitos.

Pruebas y Solución de Problemas (30 minutos):

- Anima a los estudiantes a probar sus prototipos de Smart Dustbin y a solucionar cualquier problema con el código o el hardware.
- Fomenta la experimentación con diferentes distancias de detección y ángulos del servomotor.

Día3

- Presentación del Proyecto y Discusión (60 minutos):
- Cada grupo presenta su proyecto de Smart Dustbin a la clase, explicando el diseño, los componentes y el código.
- Discute los beneficios ambientales de los sistemas inteligentes de gestión de residuos y su impacto potencial en la reducción de residuos.

Discusión Abierta y Futuras Mejoras (30 minutos):

- Facilita una discusión en clase sobre posibles mejoras en el Smart Dustbin y otras aplicaciones de tecnología similar.
- Anima a los estudiantes a hacer lluvia de ideas sobre ideas para mejorar aún más las soluciones de gestión de residuos.

Evaluaciones

Evaluar a los estudiantes en función de su participación, la funcionalidad del proyecto, su comprensión de la programación en Arduino y su capacidad para solucionar problemas.

Tarea

Asignar tarea que requiere que los estudiantes investiguen y presenten ejemplos reales de sistemas de gestión de residuos inteligentes y su impacto en la sostenibilidad.

Conclusión:

Concluye la lección resumiendo los conceptos clave aprendidos, destacando la intersección entre la tecnología y las soluciones ambientales, y alentando a los estudiantes a pensar críticamente en la aplicación de la tecnología para generar cambios positivos.

FROM GAME TO STEM

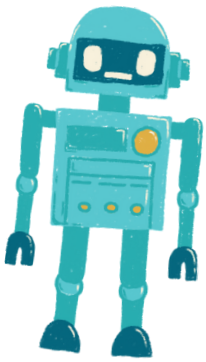
Co-funded by the Erasmus+ Programme of the European Union

Lección 10
Lección de Ecología con Arduino

Explorando la Calidad del Aire con Arduino

Duración:

1 período



Objetivo:

- Los estudiantes aprenderán sobre la importancia de la calidad del aire en la salud ambiental.
- Los estudiantes comprenderán cómo los contaminantes del aire pueden afectar a los ecosistemas y la salud humana.
- Los estudiantes construirán un sensor de calidad del aire simple basado en Arduino y recopilarán datos.
- Los estudiantes discutirán la importancia de monitorear la calidad del aire para el bienestar ecológico.

Materiales:

- Placas Arduino Uno (una por estudiante o grupo)
- Módulo de sensor de calidad del aire (por ejemplo, MQ-135)
- Cables puente
- Cables USB para la programación
- Computadoras con Arduino IDE instalado
- Proyector o pizarra para las demostraciones

Actividades

Día1

Introducción a la Calidad del Aire (10 minutos):

- Comienza la lección discutiendo la importancia del aire limpio y su impacto en los ecosistemas.
- Explica cómo los contaminantes del aire pueden afectar tanto a los entornos naturales como a la salud humana.

Sensores de Calidad del Aire (20 minutos):

- Introduce los sensores de calidad del aire y su papel en la monitorización de la contaminación del aire.
- Explica el funcionamiento básico de los sensores de calidad del aire y cómo miden varios gases.

Conceptos Básicos de Arduino (10 minutos):

- Presenta Arduino como una plataforma de microcontrolador para la recopilación de datos.
- Muestra a los estudiantes los componentes de una placa Arduino y los conceptos básicos de la escritura y carga de código.

Preparación de la Actividad Práctica (15 minutos):

- Proporciona a cada grupo una placa Arduino Uno, un módulo de sensor de calidad del aire y cables puente.
- Instruye a los estudiantes para que ensamblen el sensor de calidad del aire y lo conecten a Arduino.

Construcción y Programación del Sensor de Calidad del Aire (20 minutos):

- Guía a los estudiantes en la construcción de su sistema de sensor de calidad del aire basado en Arduino.
- Muestra a los estudiantes cómo escribir código Arduino para recopilar datos de calidad del aire del sensor.

Aquí tienes un ejemplo sencillo de código Arduino para monitorear la calidad del aire utilizando un sensor de calidad del aire MQ-135. Este código lee los datos del sensor y los muestra en el Monitor Serie. Asegúrate de tener las bibliotecas adecuadas instaladas en tu Arduino IDE, ya que el sensor MQ-135 puede requerir calibración para obtener resultados precisos.

```
// Include necessary libraries
#include <Wire.h>
#include <Adafruit_Sensor.h>
#include <Adafruit_MQ135.h>

// Define the analog pin where the MQ-135 sensor is connected
#define MQ135_PIN A0

// Create an instance of the Adafruit MQ135 class
Adafruit_MQ135 mq135(MQ135_PIN);

void setup() {
 // Initialize serial communication for data output
 Serial.begin(9600);
}

void loop() {
 // Read the MQ-135 sensor's data
 float airQuality = mq135.readCO2();

 // Print the air quality data to the Serial Monitor
 Serial.print("Air Quality: ");
 Serial.print(airQuality);
 Serial.println(" ppm");

 // Add a delay before the next reading
 delay(2000); // Adjust the delay time as needed
}
```

En este código:

- Incluimos las bibliotecas necesarias, incluyendo las bibliotecas Adafruit Sensor y Adafruit MQ135, que son comúnmente utilizadas para trabajar con el sensor de calidad del aire MQ-135. Asegúrate de tener estas bibliotecas instaladas en tu entorno de desarrollo de Arduino.
- Definimos el pin analógico (A0 en este ejemplo) al cual está conectado el sensor MQ-135 en Arduino.
- Creamos una instancia de la clase Adafruit_MQ135 para interactuar con el sensor.
- En la función setup(), inicializamos la comunicación serial para la salida de datos en el Monitor Serie.
- En la función loop(), leemos continuamente los datos de calidad del aire del sensor MQ-135 utilizando mq135.readCO2(). Los datos representan la concentración de CO2 en partes por millón (ppm).
- Los datos de calidad del aire se imprimen en el Monitor Serie, y hay un retraso de 2 segundos (ajustable) antes de tomar la próxima lectura.

Ten en cuenta que el sensor MQ-135 puede requerir calibración para obtener lecturas precisas, y el código proporcionado aquí sirve como un ejemplo básico. Dependiendo de tu aplicación específica y los requisitos de calibración, es posible que necesites ajustar el código y el proceso de calibración en consecuencia.

Recopilación de Datos y Discusión (20 minutos):

- Instruye a los estudiantes a recopilar datos de calidad del aire ejecutando sus sensores en diferentes entornos (por ejemplo, en interiores, cerca de una carretera, en un jardín).
- Pide a los estudiantes que registren y compartan sus datos con la clase.
- Lidera una discusión en clase sobre las diferencias en los datos de calidad del aire y las posibles consecuencias ecológicas.

Evaluaciones

Evalúa a los estudiantes en función de su participación, la recopilación de datos y su capacidad para discutir el impacto de la calidad del aire en los sistemas ecológicos.

Terea

Asigna tarea que requiera que los estudiantes investiguen y presenten ejemplos del mundo real de problemas ecológicos relacionados con la contaminación del aire y la importancia de la monitorización de la calidad del aire para mitigar estos problemas.

Conclusión:

Para concluir la lección, es importante resumir los conceptos ecológicos clave aprendidos y enfatizar el papel de la tecnología, en este caso Arduino, en la monitorización ambiental.