Miami Dade College

**Course Description**

**CEN2211 | C/C++ Programming for Embedded Devices | 4.00 credits**

This course teaches the principles of programming in the C/C++ languages for embedded devices. The student will learn how to create programs to control open-source hardware for building digital devices that can sense and control the physical world around them and communicate with the Internet. Prerequisite: COP1334; Corequisite: EET1033C.

**Course Competencies:**

**Competency 1:** The student will demonstrate an understanding of interaction design and physical computing by:

1. Defining Design Thinking and each of its stages
2. Explaining how to discover the user's needs and how to estimate the feasibility of a technological solution
3. Defining Minimum Viable Product
4. Solving challenges in teams, proposed by the instructor, and developing duct tape prototypes
5. Summarizing the challenges and opportunities that IoT brings to solve everyday issues
6. Explaining the importance of prototyping in interaction design
7. Comparing possible hardware and software to use in prototypes of computational things
8. Defining the hardware and software building blocks of Embedded Devices (ED)
9. Identifying suitable physical and intelligent materials and understanding their design potential
10. Researching about the similarities and differences of ED programing vs. desktop, web, and mobile programming
11. Inferring how society has been affected by the massification of ED
12. Researching about the future trends of the effect of ED in the world

**Competency 2:** The student will demonstrate an understanding of the Arduino environment, soldering and assembling electronic prototypes by:

1. Researching the Arduino environment, which is composed of the Arduino board, the Arduino IDE, and the Arduino-compatible shields together with their libraries
2. Explaining how the C and C++ languages are used to create programs that can work in the Arduino environment
3. Describing the different menus and screens in the Arduino IDE or other compatible IDEs
4. Connecting the Arduino Board to the computer and upload the program
5. Discussing all the main components, inputs, and outputs of an Arduino hardware-based solution
6. Examining the board schematic to see how they are connected
7. Using the serial monitor to read the current value of variables from the Arduino
8. Assembling an Arduino board with shields to connect with the internet, control motors, or sensors
9. Setting up the workspace and identifying all the components involved (such as wires, breadboards, multimeter, wire cutters, heat gun, soldering iron kit, disordering braid, helping hands, etc.)
10. Explaining the safety concerns when soldering
11. Performing the soldering of wires and through-hole soldering of components
12. Showing how to use the multimeter to read
13. the electrical values of a circuit

**Competency 3:** The student will demonstrate an understanding of the basic concepts of the C/C++ programming language and programming development boards by:

1. Describing an Arduino sketch's basic structure, including the setup and loop functions
2. Describing how to access the Arduino’s pins
3. from a sketch
4. Explaining the basic syntax, operators, variables, and types

5. Defining the concepts of gateways, backend, and firmware
6. Defining different development boards and listing their compatible operating systems
7. Understanding the components of a program for a development board, such as Variables, Functions, pin Mode, digital Write, delay, setup, and loop
8. Uploading and running sample programs to development boards that include digital outputs, digital inputs, analog inputs, analog outputs, if/else statements, loops, and functions
9. Modifying sample programs to adjust their parameters

**Competency 4:** The student will demonstrate an understanding of functions and control commands and loops by:

1. Writing programs (Arduino sketches) that use all the loops and control commands available in the C/C++ language
2. Writing programs that use the library function get char and putcher to read/write information, use strings, and do mathematical operations
3. Controlling programs by testing data and using logical operators
4. Writing programs that interact with users and respond to their input
5. Manipulating text with strings

**Competency 5:** The student will demonstrate an understanding of how to debug embedded software by:

1. Discussing the basic debugging requirements: controllability and observability
2. Describing how to use the UART communication protocol to gain controllability and observability
3. Explaining how to use the Serial Library to communicate with the Arduino through the serial monitor

**Competency 6:** The student will demonstrate an understanding of data management by:

1. Describing the memory access in an Arduino
2. Describing the process of filling arrays and searching values in them
3. Defining the Heap, how to allocate it, and how to free up memory
4. Identifying the components of Structures
5. Applying the proper methods for putting data in Struct Variables

**Course Competency 7:** The student will demonstrate an understanding of how to write programs that interact with sensors, actuators, displays, and speakers in embedded devices by:

1. Defining sensors and the type of information that they capture
2. Using the Pulse Width Modulation (PWM) technique for getting analog results with digital means
3. Using the analog Write () function to write an analog value (PWM wave) to a pin
4. Defining Arduino Shield
5. Listing Arduino Shields that are commonly used and identifying their applications
6. Using the Ethernet and Wi-Fi Shield libraries to connect the Arduino to the Internet
7. Defining LED displays, speakers, cameras, accelerometer, gyroscope, magnetometer, GPS receiver, switch, and temperature sensor
8. Assembling a display and a development board showing programmable content
9. Assembling an amplifier and a speaker with a developer board that can emit programmable sound
10. Assembling sensors (such as switches, cameras, proximity, and light sensors) that can be read by a program running on a development board
11. Writing programs that use the data collected by sensors and emit a response (such as text on a display or a sound) based on it

**Course Competency 8:** The student will demonstrate an understanding of how to write Arduino sketches by:

1. Programing digital and analog inputs and outputs
2. Using functions from the standard libraries
3. Interfacing with displays
4. Connecting it to the internet and configuring it as a web server
5. Storing data in EEPROM or flash memory

6. Using the Wire library to communicate with I2C / TWI devices
7. Using the Servo library allows an Arduino board to control servo motors
8. Building prototypes that, using software and hardware, can respond to the Surrounding the world with connected sensors and actuators

**Course Competency 9:** The student will demonstrate an understanding of how to interact with online services (through APIs and SDKs) and how to interact with sensors and actuators by:

1. Researching about network libraries, web services, and APIs
2. Writing a program that runs on a development board (such as an Arduino or Particle) and sends a message through the Internet (such as a Tweet using Twitter) with the current temperature (or other data collected by sensors connected to the board)
3. Writing a program that runs on a development board that takes pictures with a camera or controls a motor, and it's connected with an API

**Learning Outcomes:**

- Use quantitative analytical skills to evaluate and process numerical data
- Solve problems using critical and creative thinking and scientific reasoning
- Use computer and emerging technologies effectively