Gateway-Computer Science for Innovators and Makers
Grade 8

Curriculum Committee Members
Matt McClellan, Career and Technical Education Coordinator

Reviewed by Curriculum Advisory Committee on March 1, 2018

Reviewed by High School Business Education Teachers on May 8, 2018

Approved by the Board of Education on June 19, 2018

COURSE TITLE: PLTW Gateway – Computer Science for Innovators and Makers

GRADE LEVEL: 8th

CONTENT AREA: Career and Technical Education

Course Description

Computer Science for Innovators and Makers teaches students that programming goes beyond the virtual world into the physical world. Students are challenged to creatively use sensors and actuators to develop systems that interact with their environment. Designing algorithms and using computational thinking practices, they code and upload programs to microcontrollers that perform a variety of authentic tasks. The unit broadens students' understanding of computer science concepts through meaningful applications. Teams select and solve a personally relevant problem related to wearable technology, interactive art, or mechanical devices. *Taken from pltw.org.*

Course Rationale

Students learn about programming for the physical world by blending hardware design and software development. Using microcontrollers with inputs and outputs, they develop code that brings their physical designs to life. *Taken from pltw.org.*

Course Scope and Sequence

Unit 1: Blink! **15 class periods (90 minutes)**	**Unit 2: The Ins and Outs** **15 class periods (90 minutes)**	**Unit 3: Program the Physical World** **15 class periods (90 minutes)**

Proposed Course Materials and Resources

- Computer Lab/tablets/Chromebooks
- Inventory list provided by Project Lead the Way (www.mypltw.org)

Unit Objectives

Unit 1

The students will be able to:

1. Compare different algorithms that may be used to solve the same problem in terms of their speed, clarity, and size (e.g., different algorithms solve the same problem, but one might be faster than the other).
2. Interpret the flow of execution of algorithms and predict their outcomes.
3. Decompose a problem into parts and create solutions for each part.
4. Use an iterative design process (e.g., define the problem, generate ideas, build, test, and improve solutions) to solve problems, both independently and collaboratively.
5. Analyze the relationship between a device's computational components and its capabilities.
6. Use a systematic process to identify the source of a problem within individual and connected devices (e.g., follow a troubleshooting flow diagram, make changes to software to see if hardware will work, restart device, check connections, swap in working components).
7. Explain the processes used to collect, transform, and analyze data to solve a problem using computational tools (e.g., use an app or spreadsheet form to collect data, decide which data to use or ignore, and choose a visualization method.).
8. Represent data using different encoding schemes (e.g., binary, Unicode, Morse code, shorthand, student-created codes).
9. Explain how computer science fosters innovation and enhances nearly all careers and disciplines.
10. Describe ethical issues that relate to computing devices and networks (e.g., equity of access, security, hacking, intellectual property, copyright, Creative Commons licensing, and plagiarism).
11. Simulate how information is transmitted as packets through multiple devices over the Internet and networks.

Unit 2

The students will be able to:

1. Compare different algorithms that may be used to solve the same problem in terms of their speed, clarity, and size (e.g., different algorithms solve the same problem, but one might be faster than the other).
2. Provide proper attribution when code is borrowed or built upon.
3. Interpret the flow of execution of algorithms and predict their outcomes.
4. Design, develop, and present computational artifacts such as mobile applications that address social problems both independently and collaboratively.
5. Develop programs, both independently and collaboratively, that include sequences with nested loops and multiple branches.
6. Create variables that represent different types of data and manipulate their values.
7. Decompose a problem into parts and create solutions for each part.
8. Use an iterative design process (e.g., define the problem, generate ideas, build, test, and improve solutions) to solve problems, both independently and collaboratively.

9. Justify the hardware and software chosen to accomplish a task (e.g., comparison of the features of a tablet vs. desktop, selecting which sensors and platform to use in building a robot or developing a mobile app).
10. Analyze the relationship between a device's computational components and its capabilities.
11. Use a systematic process to identify the source of a problem within individual and connected devices (e.g., follow a troubleshooting flow diagram, make changes to software to see if hardware will work, restart device, check connections, swap in working components).
12. Explain the processes used to collect, transform, and analyze data to solve a problem using computational tools (e.g., use an app or spreadsheet form to collect data, decide which data to use or ignore, and choose a visualization method.).
13. Represent data using different encoding schemes (e.g., binary, Unicode, Morse code, shorthand, student-created codes).
14. Explain how computer science fosters innovation and enhances nearly all careers and disciplines.
15. Provide examples of how computational artifacts and devices impact health and well-being, both positively and negatively.
16. Describe ethical issues that relate to computing devices and networks (e.g., equity of access, security, hacking, intellectual property, copyright, Creative Commons licensing, and plagiarism).
17. Summarize security risks associated with weak passwords, lack of encryption, insecure transactions, and persistence of data.
18. Simulate how information is transmitted as packets through multiple devices over the Internet and networks.

Unit 3

The students will be able to:

1. Compare different algorithms that may be used to solve the same problem in terms of their speed, clarity, and size (e.g., different algorithms solve the same problem, but one might be faster than the other).
2. Provide proper attribution when code is borrowed or built upon.
3. Interpret the flow of execution of algorithms and predict their outcomes.
4. Design, develop, and present computational artifacts such as mobile applications that address social problems both independently and collaboratively.
5. Develop programs, both independently and collaboratively, that include sequences with nested loops and multiple branches.
6. Create variables that represent different types of data and manipulate their values.
7. Decompose a problem into parts and create solutions for each part.
8. Use an iterative design process (e.g., define the problem, generate ideas, build, test, and improve solutions) to solve problems, both independently and collaboratively.
9. Justify the hardware and software chosen to accomplish a task (e.g., comparison of the features of a tablet vs. desktop, selecting which sensors and platform to use in building a robot or developing a mobile app).

10. Analyze the relationship between a device's computational components and its capabilities.
11. Use a systematic process to identify the source of a problem within individual and connected devices (e.g., follow a troubleshooting flow diagram, make changes to software to see if hardware will work, restart device, check connections, swap in working components).
12. Explain the processes used to collect, transform, and analyze data to solve a problem using computational tools (e.g., use an app or spreadsheet form to collect data, decide which data to use or ignore, and choose a visualization method.).
13. Represent data using different encoding schemes (e.g., binary, Unicode, Morse code, shorthand, student-created codes).
14. Explain how computer science fosters innovation and enhances nearly all careers and disciplines.
15. Provide examples of how computational artifacts and devices impact health and well-being, both positively and negatively.
16. Describe ethical issues that relate to computing devices and networks (e.g., equity of access, security, hacking, intellectual property, copyright, Creative Commons licensing, and plagiarism).
17. Redesign a computational artifact to remove barriers to universal access (e.g., using captions on images, high contrast colors, and/or larger font sizes).

Essential Terminology/Vocabulary

Unit 1: Algorithm, algorithmic thinking, black-based coding, commands, code, debug, download, microbit, microcontroller, physical computing systems, programs, transmit, and upload.

Unit 2: Actuators, alligator clips, conductive thread, conductive paint, copper tape, electrical signals, inputs, outputs, sensor, and switch.

Unit 3: Innovator, maker, and physical computing knowledge.