

Foundations of Computer Programming Curriculum Map – Elliott Neumeister, Murphey Middle School, 2023-08-07

Middle School Computer Science courses are 9 weeks long per the curriculum, but Murphey uses an A Day – B Day system.   
As a result, my estimated times assume a total course duration of 18 weeks.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Grade Bands: 6-8** | **Unit 1** | **Unit 2** | **Unit 3** | **Unit 4** | **Unit 5** |
| **Instructional Segment:** | Employability Skills and Typing | Intro to Programming: Hedy Levels 1-10 | Intermediate Programming: Hedy Levels 11-16 | Practical Python with micro:bit | Computer Components, Hardware, and Software |
| **Estimated Time:** | 3 weeks and throughout the year | 4 weeks | 3 weeks | 4 weeks | 4 weeks |
| **Core Concepts/ Vocabulary:** | Etiquette – Practicing behaviors and speech patterns that are appropriate for business and other professional environments.  Time Management – strategies to meet deadlines, prioritize tasks, and avoid procrastination.  WPM – words per minute; a measure of typing speed. | Code – a series of instructions that are executed by a computer to solve a problem or accomplish a task.  Variable – a letter or word that stands in for another value. Frequently used for storing and recalling data in programming.  List – an ordered set of related values that can be used to store and organize data.  Conditional – an if/then statement that allows programs to behave differently based on input. | Algorithm – a procedure for solving a problem, like the steps in a recipe.  Loop – a statement that allows you to perform repetitive tasks without having to manually write each instance.  Comparison – a character like <, >, or == used to check if one value is less than, greater than, or equal to another value.  Python – a real-world programming language used by professionals in a variety of fields, from computer science to biology and physics. | Microcontroller – a small, simple computer that can execute code written on a more complex machine.  Flashing – the process of writing new code to a device or microcontroller.  Output – the results of an executed program, which may take the form of lighting LEDs, emitting a sound, displaying text or images on a screen, etc. | Hardware – physical devices that make up computers, like CPU, RAM, hard drive, monitor, mouse and keyboard.  Software – programs that run on hardware, like Windows, macOS, Microsoft Office, and Google Chrome.  Network – multiple computers connected together to exchange information, host websites, etc. |
| **GaDOE Standards:** | * **MS-CS-FCP-1** Demonstrate employability skills required by business and industry. * 1.1 Communicate effectively through writing, speaking, listening, reading, and interpersonal abilities. * 1.2 Demonstrate creativity by asking challenging questions and applying innovative procedures and methods. * 1.3 Exhibit critical thinking and problem-solving skills to locate, analyze and apply information in career planning and employment situations. * 1.4 Model work readiness traits required for success in the workplace including integrity, honesty, accountability, punctuality, time management, and respect for diversity. * 1.5 Apply the appropriate skill sets to be productive in a changing, technological, diverse workplace to be able to work independently and apply teamwork skills. * 1.6 Present a professional image through appearance, behavior, and language. * **MS-CS-FCP-6** Create digital artifacts to address a current issue requiring resolution. * 6.1 Summarize ethical, privacy, and legal issues of a digital world using current case studies. * 6.2 Collaborate as a team to develop an artifact that represents multiple perspectives regarding a global crisis. * 6.3 Analyze and explain the functionality and suitability (or appropriateness) of a computational artifact. * 6.4 Develop a program for creative expression or to satisfy personal curiosity which may have visual, audible, and/or tactile results. * 6.5 Develop a program specifically with the goal of solving a problem, creating new knowledge, or helping people, organizations, or society. | * **MS-CS-FCP-2** Explore and explain the basic components of computers and their relationships to programming. * 2.3 Demonstrate an understanding of the fundamental concepts for how computers process programming commands (hex, binary language, sequence of commands, conditional structures, looping structures). * **MS-CS-FCP-3** Utilize computational thinking to solve problems. * 3.1 Make observations and organize the concepts of modularity, including functions and methods, as it relates to programming code reusability and cloud computing in the software industry. * 3.2 Develop a working vocabulary of computational thinking including sequences, algorithms, binary, pattern matching, decomposition, abstraction, parallelization, data, automation, data collection, data analysis, Boolean, integer, branches (if...then...else), and iteration {loops (For, While)}. * 3.3 Analyze the problem-solving process, the input-process-output-storage model of a computer, and how computers help humans solve problems. * 3.4 Develop an algorithm to decompose a problem of a daily task. | * **MS-CS-FCP-3** Utilize computational thinking to solve problems. * 3.1 Make observations and organize the concepts of modularity, including functions and methods, as it relates to programming code reusability and cloud computing in the software industry. * 3.2 Develop a working vocabulary of computational thinking including sequences, algorithms, binary, pattern matching, decomposition, abstraction, parallelization, data, automation, data collection, data analysis, Boolean, integer, branches (if...then...else), and iteration {loops (For, While)}. * 3.3 Analyze the problem-solving process, the input-process-output-storage model of a computer, and how computers help humans solve problems. * 3.4 Develop an algorithm to decompose a problem of a daily task. * **MS-CS-FCP-4** Design, develop, debug and implement computer programs. * 4.1 Develop a working vocabulary of programming including flowcharting and/or storyboarding, coding, debugging, user interfaces, usability, variables, lists, loops, conditionals, programming language, and events. * 4.2 Utilize the design process to brainstorm, implement, test, and revise an idea. * 4.3 Cite evidence on how computers represent data and media (sounds, images, video, etc.). * 4.4 Design a user interface and test with other users using a paper prototype. * 4.5 Implement a simple algorithm in a computer program. * 4.6 Develop an event driven program. * 4.7 Create a program that accepts user and/or sensor input and stores the result in a variable. * 4.8 Create a computer program that implements a loop. * 4.9 Develop a program that makes a decision based on data or user input. * 4.10 Debug a program with an error. | * **MS-CS-FCP-5** Explore the relationship between computer hardware and software. * 5.2 Investigate how software interacts with hardware in the boot process. * 5.3 Analyze and explain how computers communicate information with simple hardware inputs and outputs. * 5.4 Create a product that analyzes how simple computer hardware can be used to develop innovative new products that interact with the physical world. * 5.5 Design a computer program that senses something in the real world and changes an output based on the input. * **MS-CS-FCP-6** Create digital artifacts to address a current issue requiring resolution. * 6.1 Summarize ethical, privacy, and legal issues of a digital world using current case studies. * 6.2 Collaborate as a team to develop an artifact that represents multiple perspectives regarding a global crisis. * 6.3 Analyze and explain the functionality and suitability (or appropriateness) of a computational artifact. * 6.4 Develop a program for creative expression or to satisfy personal curiosity which may have visual, audible, and/or tactile results. * 6.5 Develop a program specifically with the goal of solving a problem, creating new knowledge, or helping people, organizations, or society. | * **MS-CS-FCP-2** Explore and explain the basic components of computers and their relationships to programming. * 2.1 Identify the basic components of the computer (processor, operating system, memory, storage, ethernet ports) by disassembling and reassembling a demonstration model personal computer (may be done ‘virtually’ online if demo model is not available). * 2.2 Demonstrate an understanding of key functional components (input/output devices, software applications, wi-fi and/or Ethernet, and IP addresses). * 2.3 Demonstrate an understanding of the fundamental concepts for how computers process programming commands (hex, binary language, sequence of commands, conditional structures, looping structures). * **MS-CS-FCP-5** Explore the relationship between computer hardware and software. * 5.1 Develop a working vocabulary of embedded computing including digital, analog, events, microcontrollers, sensors, light emitting diodes (LED), switches, servos, cloud computing, and internet of things. * 5.2 Investigate how software interacts with hardware in the boot process. * 5.3 Analyze and explain how computers communicate information with simple hardware inputs and outputs. * 5.4 Create a product that analyzes how simple computer hardware can be used to develop innovative new products that interact with the physical world. * 5.5 Design a computer program that senses something in the real world and changes an output based on the input. |