Computer Science for Innovators & Makers
Project Lead the Way

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.

Curriculum Framework - Lesson A

Curriculum Framework - Lesson B

Lesson 1 - Blink!

Overview: Students begin to explore the capabilities of physical computing systems with The Digital Dive game, an engaging, live-action activity where students "become" computer parts and transmit commands. They learn to use algorithmic thinking as they prepare to code. Students use block-based coding to create, download, and upload programs to the micro:bit microcontroller. They learn processes and gain skills to debug programs starting with pre-bugged programs. They apply these skills to their own project where they code a blinking message that includes text, images such as emojis, and animation.

ESSENTIAL QUESTIONS: ***Students will keep considering …***

Q 1 - How is a design process used to develop physical computing systems?

Q 2 - What do programming best practices look like?

Priority Competencies:	**Priority Competencies:**
Transportable Knowledge and Skills: COM-A.2 Communicate to meet the needs of the audience and be appropriate to the situation. COM-A.3 Document work, including processes, research, and	**Technical Knowledge and Skills:**

solutions.	
Supporting Competencies: **Transportable Knowledge and Skills** CAE-A.1 Explore a variety of careers related to engineering, biomedical sciences, and computer science. CAE-A.2 Identify skills that are needed for a variety of careers (such as communication and collaboration). CAE-A.3 Explore and reflect on your personal interests and strengths in relation to diverse career opportunities. COM-A.1 Use accurate and appropriate terminology. COM-A.4 Use reliable evidence to support a claim. COL-A.1 Demonstrate successful collaboration through effective communication and constructive feedback. COL-A.2 Apply team norms to encourage productivity and define how a team will function and measure its success. ERM-A.1 Analyze ethical considerations and their impact in decision making. CCP-A.2 Identify appropriate design requirements (criteria and constraints). CCP-A.3 Generate ideas or build upon other ideas to innovate. CCP-A.5 Iteratively design and develop a solution. CCP-C.1 Create and follow a plan to solve a problem. CCP-C.2 Decompose a problem into smaller parts. CCP-C.4 Create a step-by-step process to complete a task. CCP-D.1 Dissect a product to gain understanding about its functionality.	**Supporting Competencies:** **Technical Knowledge and Skills:** AAP-A.1 Analyze, break down, and explain the logic of an algorithm. AAP-A.2 Create simple algorithms that involve variables, conditionals, operators, or logic. AAP-B.1 Identify and describe the high-level structures of a program, such as user interface components, data components, event handlers, and procedures. AAP-C.1 Plan a program using appropriate strategies (such as flowcharting). AAP-C.2 Create appropriate event handlers to respond to runtime events. AAP-C.3 Apply programming best practices, such as creating or improving documentation, using descriptive variables and procedure names, using comments, and testing code frequently. AAP-C.4 Debug programs or identify hardware issues. AAP-D.1 Find code relevant to meet a need and extend or apply it to a new purpose. CSY-A.1 Identify how a user interacts with the parts of a computational system and how these parts interact with each other. CSY-A.2 Select and justify the hardware chosen to accomplish a task. SIOC-A.1 Identify the positive and negative ways that computing has changed or is changing the way people work, live, and play.
Pacing Calendar **At A Glance** **Lesson 1 Assessment**	

Lesson 2 - The Ins and Outs

Overview: In this lesson, students explore a variety of sensors and actuators to use as inputs and outputs in physical computing projects. Using different materials to transfer electrical signals, such as conductive thread, alligator clips, conductive paint, and copper tape, students create their own input device—a sensor or switch—to interact with a program they develop on the microcontroller. They use these skills in the lesson's project to design, develop, and program a system to protect safes and secrets.

ESSENTIAL QUESTIONS: *Students will keep considering …*

Q1 - How can algorithmic thinking skills be used across multiple disciplines?

Q2 - How can computer programs solve problems?

Priority Competencies:

Transportable Knowledge and Skills:
COM-A.1 Use accurate and appropriate terminology.
COM-A.2 Communicate to meet the needs of the audience and be appropriate to the situation.
COM-A.3 Document work, including processes, research, and solutions.
COL-A.1 Demonstrate successful collaboration through effective communication and constructive feedback.
COL-A.2 Apply team norms to encourage productivity and define how a team will function and measure its success.
CCP-A.2 Identify appropriate design requirements (criteria and constraints).
CCP-A.3 Generate ideas or build upon other ideas to innovate.
CCP-C.1 Create and follow a plan to solve a problem.
CCP-C.2 Decompose a problem into smaller parts.
CCP-C.4 Create a step-by-step process to complete a task.
CCP-C.5 Create and execute a plan to manage and use resources (such as time, materials, tools).

Priority Competencies:

Technical Knowledge and Skills:
AAP-A.1 Analyze, break down, and explain the logic of an algorithm.
AAP-A.2 Create simple algorithms that involve variables, conditionals, operators, or logic.
AAP-B.1 Identify and describe the high-level structures of a program, such as user interface components, data components, event handlers, and procedures.
AAP-C.1 Plan a program using appropriate strategies (such as flowcharting).
AAP-C.2 Create appropriate event handlers to respond to runtime events.
AAP-C.3 Apply programming best practices, such as creating or improving documentation, using descriptive variables and procedure names, using comments, and testing code frequently.
AAP-C.4 Debug programs or identify hardware issues.
AAP-D.1 Find code relevant to meet a need and extend or apply it to a new purpose.
CSY-A.3 Assemble, wire, and program a system to complete a task.

Supporting Competencies: **Transportable Knowledge and Skills** CAE-A.1 Explore a variety of careers related to engineering, biomedical sciences, and computer science. CAE-A.2 Identify skills that are needed for a variety of careers (such as communication and collaboration). CAE-A.3 Explore and reflect on your personal interests and strengths in relation to diverse career opportunities. COL-A.3 Identify and evaluate positive and negative behaviors that impact the team's effectiveness. COL-A.4 Describe one's individual role and expectations of performance within the team and support other team members, if needed, to meet team goals. CCP-A.1 Describe major steps of a design process and the typical tasks involved in each step. CCP-A.4 Evaluate solution ideas against the design requirements and justify the best solution to pursue. CCP-A.5 Iteratively design and develop a solution. CCP-A.6 Develop and implement a plan to test and evaluate a potential solution to verify that it best meets all design requirements. CCP-B.1 Investigate the types of interactions between users and a proposed solution. CCP-B.2 Explain the importance of involving prospective users early and often during the design process. CCP-B.4 Incorporate safety in all designs, products, and solutions. CCP-B.5 Design solutions to optimize the user experience. CCP-C.3 Collect, display, analyze, and interpret data (such as diagrams, charts, graphs, and tables) to draw a conclusion.	**Supporting Competencies:** **Technical Knowledge and Skills:** DAT-A.1 Store, access, and update data stored in variables or lists. DAT-A.2 Trace a program and deduce the values that variables or loops will have after the code is executed. CSY-A.1 Identify how a user interacts with the parts of a computational system and how these parts interact with each other. CSY-A.2 Select and justify the hardware chosen to accomplish a task. SIOC-A.1 Identify the positive and negative ways that computing has changed or is changing the way people work, live, and play.

CCP-D.2 Describe how the functionality of a product changes depending on how it is used.	
Pacing Calendar **At A Glance** **Lesson 2 Assessment**	

Lesson 3 - Program the Physical World	
Overview: Within teams, students become innovators and makers. Teams apply their physical computing knowledge and skills as they design and create one of three problem options: ● A wearable safety device someone might use when completing a physical activity outside at night ● An engaging art installation to help improve a community space ● A useful mechanical dispenser for a person or animal who needs assistance to retrieve an object Teams collaborate and learn that solving authentic problems involves the unit content knowledge, as well as skills from other disciplines, such as communications, mathematics, and science.	
ESSENTIAL QUESTIONS: ***Students will keep considering …*** **Q1 –How can algorithmic thinking skills be used across multiple disciplines?** **Q2 - How can computer programs solve problems?** **Q3 –How do you express yourself and your creativity through computer science?**	
Priority Competencies: **Transportable Knowledge and Skills:** COM-A.1 Use accurate and appropriate terminology. COM-A.2 Communicate to meet the needs of the audience and be appropriate to the situation. COM-A.3 Document work, including processes, research, and solutions.	**Priority Competencies:** **Technical Knowledge and Skills:** DAT-A.2 Trace a program and deduce the values that variables or loops will have after the code is executed. AAP-A.1 Analyze, break down, and explain the logic of an algorithm.

COL-A.1 Demonstrate successful collaboration through effective communication and constructive feedback.	AAP-C.3 Apply programming best practices, such as creating or improving documentation, using descriptive variables and procedure names, using comments, and testing code frequently. AAP-E.1 Identify how abstraction hides the complexity of a task.
Pacing Calendar **Summative Assessment** **At A Glance**	

Supporting Competencies: **Transportable Knowledge and Skills** CAE-A.1 Explore a variety of careers related to engineering, biomedical sciences, and computer science. CAE-A.2 Identify skills that are needed for a variety of careers (such as communication and collaboration). CAE-A.3 Explore and reflect on your personal interests and strengths in relation to diverse career opportunities. COL-A.2 Apply team norms to encourage productivity and define how a team will function and measure its success. COL-A.3 Identify and evaluate positive and negative behaviors that impact the team's effectiveness. COL-A.4 Describe one's individual role and expectations of performance within the team and support other team members, if needed, to meet team goals. CCP-A.1 Describe major steps of a design process and the typical tasks involved in each step. CCP-A.2 Identify appropriate design requirements (criteria and constraints). CCP-A.3 Generate ideas or build upon other ideas to innovate. CCP-A.4 Evaluate solution ideas against the design requirements and justify the best solution to pursue. CCP-A.5 Iteratively design and develop a solution. CCP-A.6 Develop and implement a plan to test and evaluate a potential solution to verify that it best meets all design requirements. CCP-B.1 Investigate the types of interactions between users and a proposed solution. CCP-C.1 Create and follow a plan to solve a problem. CCP-C.2 Decompose a problem into smaller parts. CCP-B.2 Explain the importance of involving prospective users early and often during the design process. CCP-B.3 Consider accessibility and equity when designing and creating solutions.	**Supporting Competencies:** **Technical Knowledge and Skills:** DAT-A.1 Store, access, and update data stored in variables or lists. AAP-A.2 Create simple algorithms that involve variables, conditionals, operators, or logic. AAP-B.1 Identify and describe the high-level structures of a program, such as user interface components, data components, event handlers, and procedures. AAP-C.1 Plan a program using appropriate strategies (such as flowcharting). AAP-C.2 Create appropriate event handlers to respond to runtime events. AAP-C.4 Debug programs or identify hardware issues. AAP-D.1 Find code relevant to meet a need and extend or apply it to a new purpose. CSY-A.1 Identify how a user interacts with the parts of a computational system and how these parts interact with each other. CSY-A.2 Select and justify the hardware chosen to accomplish a task. CSY-A.3 Assemble, wire, and program a system to complete a task. SIOC-A.1 Identify the positive and negative ways that computing has changed or is changing the way people work, live, and play.

CCP-B.4 Incorporate safety in all designs, products, and solutions. CCP-B.5 Design solutions to optimize the user experience. CCP-C.1 Create and follow a plan to solve a problem. CCP-C.2 Decompose a problem into smaller parts. CCP-C.3 Collect, display, analyze, and interpret data (such as diagrams, charts, graphs, and tables) to draw a conclusion. CCP-C.4 Create a step-by-step process to complete a task. CCP-C.5 Create and execute a plan to manage and use resources (such as time, materials, tools). CCP-D.1 Dissect a product to gain understanding about its functionality.	

Pacing Calendar

At A Glance