

Grade level: 9-12

Course Title: Computer Science Essentials

**Topic/Concept: Creative Computing:
Building with Blocks**

Time Allotment: 9 weeks

Unit Sequence: 1

Major Concepts to be learned:

- Students work with MIT App Inventor to create basic apps that rely on the concepts of event-driven programming, branching, iteration, variables, and abstraction—the building blocks of creating with code. Students are introduced to essential computational thinking practices, such as developing abstractions, collaborating around computing, and communicating as they create, test, and refine computational artifacts of Android/Apple apps.

Expected Skills to be demonstrated:

- **Program Design and Development**
Create programs using procedural and event-driven paradigms
Design a user interface based on human-computer interaction principles
Debug and test code
- **Interpret Documentation**
Create task reflection with the Interpreted Performance Guide
Create task reflection with the Interpreted Performance Guide Response Template
Use application programming interfaces (APIs)
- **Programming Languages**
MIT App Inventor based on Blockly
- **Tools and Software**
MIT App Inventor based on Blockly
- **Professional Skills**
Pair Programming
Accountability to a team
Collaboration on design and implementation
Presentation/Communication
Ethics

PA Standards/Anchors:

3A-CS-01 Explain how abstractions hide the underlying implementation details of computing systems embedded in everyday objects.
3A-CS-02 Compare levels of abstraction and interactions between application software, system software, and hardware layers.
3A-CS-03 Develop guidelines that convey systematic troubleshooting strategies that others can use to identify and fix errors
3A-NI-05 Give examples to illustrate how sensitive data can be affected by malware and other attacks.
3A-NI-06 Recommend security measures to address various scenarios based on factors such as efficiency, feasibility, and ethical impacts.
3A-DA-10 Evaluate the tradeoffs in how data elements are organized and where data is stored.
3A-AP-13 Create prototypes that use algorithms to solve computational problems by leveraging prior student knowledge and personal interests.
3A-AP-14 Use lists to simplify solutions, generalizing computational problems instead of repeatedly using simple variables.
3A-AP-15 Justify the selection of specific control structures when tradeoffs involve implementation, readability, and program performance, and explain the benefits and drawbacks of choices made.
3A-AP-16 Design and iteratively develop computational artifacts for practical intent, personal expression, or to address a societal issue by using events to initiate instructions.
3A-AP-17 Decompose problems into smaller components through systematic analysis, using constructs such as procedures, modules, and/or objects.
3A-AP-18 Create artifacts by using procedures within a program, combinations of data and procedures, or independent but interrelated programs.
3A-AP-19 Systematically design and develop programs for broad audiences by incorporating feedback from users.
3A-AP-21 Evaluate and refine computational artifacts to make them more usable and accessible.
3A-AP-22 Design and develop computational artifacts working in team roles using collaborative tools.
3A-AP-23 Document design decisions using text, graphics, presentations, and/or demonstrations in the development of complex programs.
3A-IC-24 Evaluate the ways computing impacts personal, ethical, social, economic, and cultural practices.
3A-IC-25 Test and refine computational artifacts to reduce bias and equity deficits.
3A-IC-26 Demonstrate ways a given algorithm applies to problems across disciplines
3A-IC-27 Use tools and methods for collaboration on a project to increase connectivity of people in different cultures and career fields.
3A-IC-30 Evaluate the social and economic implications of privacy in the context of safety, law, or ethics.

Instructional Strategies:

Assessments:

- | | |
|---|--|
| <ul style="list-style-type: none">• Activities• Projects• Teacher Observation | |
|---|--|

Grade level: 9-12

Course Title: Computer Science Essentials

**Topic/Concept: Creative Computing:
Building with Blocks**

Time Allotment: 9 weeks

Unit Sequence: 2

Major Concepts to be learned:

- Unit 2 continues to reinforce coding fundamentals as students are gradually introduced to text-based programming. In this unit, students explore the impacts of computer science on our society and bring coding off the screen and into the physical world. Students learn how images can be used to make decisions in programs and explore real-world applications and innovations that will shape our future.

Expected Skills to be demonstrated:

- **Program Design and Development**
Create programs using procedural and event-driven paradigms
Use image processing to control program flow and vehicle movement
Debug and test code
- **Interpret Documentation**
Create task reflection with the Interpreted Performance Guide
Create task reflection with the Interpreted Performance Guide Response Template
- **Programming Languages**
VEXcode Blocks
- **Tools and Software**
VEXcode Blocks
Trinkets
- **Professional Skills**
Pair Programming
Agile Project Development/Scrum
Accountability to a team
Collaboration on design and implementation
Presentation/Communication
Ethics

PA Standards/Anchors:

3A-CS-01 Explain how abstractions hide the underlying implementation details of computing systems embedded in everyday objects.
3A-CS-03 Develop guidelines that convey systematic troubleshooting strategies that others can use to identify and fix errors
3A-DA-10 Evaluate the tradeoffs in how data elements are organized and where data is stored.
3A-AP-13 Create prototypes that use algorithms to solve computational problems by leveraging prior student knowledge and personal interests.
3A-AP-16 Design and iteratively develop computational artifacts for practical intent, personal expression, or to address a societal issue by using events to initiate instructions.
3A-AP-17 Decompose problems into smaller components through systematic analysis, using constructs such as procedures, modules, and/or objects.
3A-AP-18 Create artifacts by using procedures within a program, combinations of data and procedures, or independent but interrelated programs.
3A-AP-21 Evaluate and refine computational artifacts to make them more usable and accessible.
3A-AP-22 Design and develop computational artifacts working in team roles using collaborative tools.
3A-AP-23 Document design decisions using text, graphics, presentations, and/or demonstrations in the development of complex programs.
3A-IC-24 Evaluate the ways computing impacts personal, ethical, social, economic, and cultural practices.

Instructional Strategies:

<ul style="list-style-type: none">• Activities• Projects• Teacher Observation	
---	--

Assessments:

Grade level: 9-12

Course Title: Computer Science Essentials

**Topic/Concept: Creative Computing:
Building with Blocks**

Time Allotment: 11 weeks

Unit Sequence: 3

Major Concepts to be learned:

- Unit 3 is for students to begin to understand and use the flexibility and power of programming in a text-based environment. In this unit, students continue to build on coding fundamentals as they apply the same coding concepts, computational thinking practices, and development processes introduced in units 1 and 2.

Expected Skills to be demonstrated:

- **Program Design and Development**
Create programs using procedural and event-driven paradigms
Design a user interface (UI) based on human-computer interaction (HCI) principles
Use image processing to control program flow and vehicle movement
Debug and test code
- **Interpret Documentation**
Create task reflection with the Interpreted Performance Guide
Create task reflection with the Interpreted Performance Guide Response Template
Use application programming interfaces (APIs)
- **Programming Languages**
MIT App Inventor based on Blockly
VEXcode Blocks
Python
- **Tools and Software**
MIT App Inventor based on Blockly
VEXcode Blocks
Trinkets
- **Professional Skills**
Pair Programming
Agile Project Development/Scrum
Accountability to a team
Collaboration on design and implementation
Presentation/Communication
Public Speaking
Ethics

PA Standards/Anchors:

3A-CS-01 Explain how abstractions hide the underlying implementation details of computing systems embedded in everyday objects.
3A-CS-02 Compare levels of abstraction and interactions between application software, system software, and hardware layers.
3A-CS-03 Develop guidelines that convey systematic troubleshooting strategies that others can use to identify and fix errors
3A-DA-10 Evaluate the tradeoffs in how data elements are organized and where data is stored.
3A-AP-13 Create prototypes that use algorithms to solve computational problems by leveraging prior student knowledge and personal inter
3A-AP-14 Use lists to simplify solutions, generalizing computational problems instead of repeatedly using simple variables.
3A-AP-15 Justify the selection of specific control structures when tradeoffs involve implementation, readability, and program performance, explain the benefits and drawbacks of choices made.
3A-AP-16 Design and iteratively develop computational artifacts for practical intent, personal expression, or to address a societal issue by u events to initiate instructions.
3A-AP-17 Decompose problems into smaller components through systematic analysis, using constructs such as procedures, modules, and/o objects.
3A-AP-18 Create artifacts by using procedures within a program, combinations of data and procedures, or independent but interrelated programs.
3A-AP-19 Systematically design and develop programs for broad audiences by incorporating feedback from users.
3A-AP-21 Evaluate and refine computational artifacts to make them more usable and accessible.
3A-AP-22 Design and develop computational artifacts working in team roles using collaborative tools.
3A-AP-23 Document design decisions using text, graphics, presentations, and/or demonstrations in the development of complex programs.
3A-IC-24 Evaluate the ways computing impacts personal, ethical, social, economic, and cultural practices.

Instructional Strategies:

<ul style="list-style-type: none">• Activities• Projects• Teacher Observation

Assessments:

--

Grade level: 9-12

Course Title: Computer Science Essentials

**Topic/Concept: Creative Computing:
Building with Blocks**

Time Allotment: 11 weeks

Unit Sequence: 4

Major Concepts to be learned:

- Unit 4 in CSE allows students to apply all that they have learned in a student-defined, student-driven development. Whether creating an app, a website, or a physical computing device, students apply computational thinking practices and a strategic development process to create computational artifacts that solve problems and create value for others. Students collaborate the way computing professionals do as they pursue solutions to authentic needs. For those students continuing to PLTW CSP, this unit provides an excellent model of how to participate in, document, and create a performance task for AP CSP.

Expected Skills to be demonstrated:

- **Program Design and Development**
Create programs using procedural and event-driven paradigms
Design a user interface (UI) based on human-computer interaction (HCI) principles
Use image processing to control program flow and vehicle movement
Debug and test code
- **Interpret Documentation**
Create task reflection with the Interpreted Performance Guide
Create task reflection with the Interpreted Performance Guide Response Template
Use application programming interfaces (APIs)
- **Programming Languages**
MIT App Inventor based on Blockly
VEXcode Blocks
Python
- **Tools and Software**
MIT App Inventor based on Blockly
VEXcode Blocks
Trinkets
- **Professional Skills**
Pair Programming
Agile Project Development/Scrum
Accountability to a team
Collaboration on design and implementation
Presentation/Communication
Public Speaking
Ethics

PA Standards/Anchors:

3A-CS-01 Explain how abstractions hide the underlying implementation details of computing systems embedded in everyday objects.

3A-AP-13 Create prototypes that use algorithms to solve computational problems by leveraging prior student knowledge and personal interest.

3A-AP-15 Justify the selection of specific control structures when tradeoffs involve implementation, readability, and program performance, explain the benefits and drawbacks of choices made.

3A-AP-16 Design and iteratively develop computational artifacts for practical intent, personal expression, or to address a societal issue by using events to initiate instructions.

3A-AP-17 Decompose problems into smaller components through systematic analysis, using constructs such as procedures, modules, and/or objects.

3A-AP-18 Create artifacts by using procedures within a program, combinations of data and procedures, or independent but interrelated programs.

3A-AP-19 Systematically design and develop programs for broad audiences by incorporating feedback from users.

3A-AP-21 Evaluate and refine computational artifacts to make them more usable and accessible.

3A-AP-22 Design and develop computational artifacts working in team roles using collaborative tools.

3A-AP-23 Document design decisions using text, graphics, presentations, and/or demonstrations in the development of complex programs.

3A-IC-25 Test and refine computational artifacts to reduce bias and equity deficits.

3A-IC-27 Use tools and methods for collaboration on a project to increase connectivity of people in different cultures and career fields.

Instructional Strategies:

Assessments:

- | | |
|---|--|
| <ul style="list-style-type: none">• Activities• Projects• Teacher Observation | |
|---|--|