Unit 4: User-Defined Functions and Structured Programming

Content Area: **Technology**
Course(s): **Computer Programming C++ I & II**
Time Period: **8 weeks**
Length: **Weeks**
Status: **Published**

Unit Overview

In this unit students will learn to read with understanding and write programs involving user-defined functions and learn how to apply the principles of Structured Programming. Students will be able to take a description of a problem or algorithm and break it down into simpler tasks that can be coded as functions, building on prior skills of analyzing algorithms for their data content and structures. Students will know the syntax for defining and calling functions and be able to apply this knowledge to analyzing existing programs and writing their own. Students will know how to write parameterless functions as well as functions that accept parameters, and will understand the difference between parameters passed by reference and parameters passed by value. Students will know how to write both void-returning and value-returning functions. Students will gain experience using objects defined in the C++ Standard Library and member function calls on objects. Special emphasis will be given to documenting the use and purpose of functions as well as testing and correcting errors.

Transfer

At the completion of this unit students should be able to write and use user-defined functions and know how to apply the principles of Structured Programming. Students should be able to take a problem description of intermediate difficulty, break it down into functions and data structures, and write, test and debug code that implements an appropriate algorithm.

Meaning

Understandings

-User-defined functions help break a complex task down into simpler, more manageable pieces

-Programming tasks need to be analyzed to determine which parts of an algorithm can profitably be expressed as separate functions

-Functions are declared separately from other code (not inside other functions), starting with the return type, name of the function, and a parameter list in parentheses, followed by the body of the function in curly braces

-Functions that return a value need to exit with a return statement

-Void functions can exit early with an 'empty' return statement

-Functions are called by naming the function followed by a comma-separated list of actual parameters enclosed in parentheses

-User-defined functions should have descriptive names and clear documentation of their use and purpose

-Function parameters that are passed by value are copied when the function is called, and changes made to the copy cannot affect variables outside that function

-Parameters that are passed by reference affect the original copy of a variable

-Formal parameters appear in a function definition, while actual parameters appear at the site of a function call

-Objects (such as file stream objects) have member functions that can be called by naming the object, applying the dot operator, and then naming the function and parameter list as with ordinary functions

Essential Questions

-What are the benefits of breaking a programming task into separate user-defined functions? What are the circumstances in which it is beneficial to do so?

-How are value-returning functions and void-returning functions different? How will I know which to use in any given circumstance?

-When is it appropriate to use pass-by-reference parameters versus pass-by-value parameters when writing functions?

-How are function calls different from unconditional branches (goto)?

-When is it appropriate to use an early return from a function?

-How should I document the use and purpose of functions I've written?

-What are the steps required for reading and writing to files using the C++ standard library?

Application of Knowledge and Skill

Students will know...

-user-defined functions help break a complex task down into simpler, more manageable pieces

-programming tasks need to be analyzed to determine which parts of an algorithm can profitably be expressed as separate functions

-functions are declared separately from other code (not inside other functions), starting with the return type, name of the function, and a parameter list in parentheses, followed by the body of the function in curly braces

-functions that return a value need to exit with a return statement

-void functions can exit early with an 'empty' return statement

-functions are called by naming the function followed by a comma-separated list of actual parameters enclosed in parentheses

-user-defined functions should have descriptive names and clear documentation of their use and purpose

-function parameters that are passed by value are copied when the function is called, and changes made to the copy cannot affect variables outside that function

-parameters that are passed by reference affect the original copy of a variable

-formal parameters appear in a function definition, while actual parameters appear at the site of a function call

Students will be skilled at...

-defining and calling functions with proper syntax, including return type, descriptive name, formal parameter list, and properly indented function body, as well as descriptive documentation

-analyzing which parts of a simple algorithm should be factored out as separate functions

-writing and using value-returning functions to replace a frequently-needed value of a calculation

-writing void functions that accept parameters to replace a common pattern of code

-writing functions that accept pass-by-value or pass-by-reference parameters as appropriate

-using the C++ standard library to read and write to and from files

-explaining the differences between formal and actual parameters

-calling member functions on objects (e.g. file stream objects for file i/o)

Academic Vocabulary

function parameter, formal parameter, actual parameter, pass-by-value, pass-by-reference, void keyword, return keyword, function call, structured programming, software library, structured programming

Learning Goal 1

Students will be able to read with understanding as well as write functions, including void-returning and value-returning functions with or without parameters, and compare and contrast pass-by-value versus pass-by-reference parameters.

Target 1

Students will be able to write void-returning functions, with and without parameters, to perform simple repeated programming tasks, and differentiate programming problems that can be solved with looping from those that can be solved by writing a void-returning function.

Additional Standards:

9.3.IT-PRG.1,2,3

TECH.8.2.12.E	Computational Thinking: Programming: Computational thinking builds and enhances problem solving, allowing students to move beyond using knowledge to creating knowledge.
TECH.8.2.12.E.3	Use a programming language to solve problems or accomplish a task (e.g., robotic functions, website designs, applications, and games).
TECH.8.2.12.E.CS1	Computational thinking and computer programming as tools used in design and engineering.

Target 2

Students will be able to write simple functions that return a value in the context of a programming task, and identify and correct errors in their own and other's code.

Additional Standards:

9.3.IT-PRG.4,5

TECH.8.2.12.E.3	Use a programming language to solve problems or accomplish a task (e.g., robotic functions, website designs, applications, and games).
TECH.8.2.12.E.4	Use appropriate terms in conversation (e.g., troubleshooting, peripherals, diagnostic software, GUI, abstraction, variables, data types and conditional statements).
TECH.8.2.12.E.CS1	Computational thinking and computer programming as tools used in design and engineering.

Target 3

Students will be able to write functions using both pass-by-reference and pass-by-value parameters, and compare and contrast the applicability of each to common programming scenarios.

Additional Standards:

9.3.IT-PRG.6, 9.3.ST.6

TECH.8.2.12.E.1	Demonstrate an understanding of the problem-solving capacity of computers in our world.
TECH.8.2.12.E.3	Use a programming language to solve problems or accomplish a task (e.g., robotic functions, website designs, applications, and games).

Learning Goal 2

Students will be able to make use of the functions and classes in the C++ standard library to perform common useful tasks, such as file input/output, random number generation and string searching and replacing, and use them to solve programming problems.

Target 1

Students will be able to write to and read from a file using the C++ standard library and filestream objects and solve programming problems involving persistent data storage and retrieval.

Additional Standards:

9.3.ST-ET.1, 3

TECH.8.1.12.C.CS1	Interact, collaborate, and publish with peers, experts, or others by employing a variety of digital environments and media.
TECH.8.1.12.C.CS2	Communicate information and ideas to multiple audiences using a variety of media and formats.
TECH.8.1.12.C.CS4	Contribute to project teams to produce original works or solve problems.
TECH.8.1.12.F.CS2	Plan and manage activities to develop a solution or complete a project.
TECH.8.2.12.E	Computational Thinking: Programming: Computational thinking builds and enhances problem solving, allowing students to move beyond using knowledge to creating knowledge.
TECH.8.2.12.E.3	Use a programming language to solve problems or accomplish a task (e.g., robotic functions, website designs, applications, and games).
TECH.8.2.12.E.4	Use appropriate terms in conversation (e.g., troubleshooting, peripherals, diagnostic software, GUI, abstraction, variables, data types and conditional statements).

Target 2

Students will be able to parse (analyze and process) the input from a file using string operations in conjunction with filestream objects, and identify and correct logical and syntax errors in their own and other's programs.

Additional Standards:

9.3.ST-ET.4, 5

TECH.8.2.12.C.4	Explain and identify interdependent systems and their functions.
TECH.8.2.12.E	Computational Thinking: Programming: Computational thinking builds and enhances problem solving, allowing students to move beyond using knowledge to creating knowledge.
TECH.8.2.12.E.2	Analyze the relationships between internal and external computer components.
TECH.8.2.12.E.3	Use a programming language to solve problems or accomplish a task (e.g., robotic functions, website designs, applications, and games).

Learning Goal 3

Students will be able to explain the impact of unethical use of computer technology, including the consequences of the hacking of customer and other sensitive information, as well as evaluate the potential to address these vulnerabilities with computer security measures.

Target 1

Students will be able to clearly differentiate ethical and unethical use of computers to access information.

Additional Standards:

9.3.IT.4, 9.3.ST-SM.3

TECH.8.1.12.E.1	Produce a position statement about a real world problem by developing a systematic plan of investigation with peers and experts synthesizing information from multiple sources.
TECH.8.1.12.E.2	Research and evaluate the impact on society of the unethical use of digital tools and present your research to peers.
TECH.8.1.12.F.CS1	Identify and define authentic problems and significant questions for investigation.

Target 2

Students will be able to compare various types of computer security vulnerabilities and evaluate the measures that can be taken by individuals and organizations to address those vulnerabilities.

TECH.8.1.12.E.2	Research and evaluate the impact on society of the unethical use of digital tools and present your research to peers.
TECH.8.1.12.E.CS1	Plan strategies to guide inquiry.
TECH.8.2.12.D.6	Synthesize data, analyze trends and draw conclusions regarding the effect of a technology on the individual, society, or the environment and publish conclusions.

Summative Assessment

Chapter and benchmark assessments

Project-based assessments, including:

-Bokmon (Pac-man clone)

-Casino games (betting strategies)

-Blackjack or another card game

21st Century Life and Careers

CAEP.9.2.12.C.1	Review career goals and determine steps necessary for attainment.
CAEP.9.2.12.C.7	Examine the professional, legal, and ethical responsibilities for both employers and employees in the global workplace.
CAEP.9.2.12.C.8	Assess the impact of litigation and court decisions on employment laws and practices.

Formative Assessment and Performance Opportunities

Daily challenge tasks, individual and group project presentations, portfolio assessments, and in-class observation, demonstration and questioning.

Accommodations/Modifications

- Supplemental project guides for extra help

-One-on-one assistance

-Extra challenge for projects
-Adding pathfinding for the ghosts in Bokmon

-Enhancing the card game with visuals

Unit Resources

-Bokmon, Card game, etc. project guides

-www.cplusplus.com

-Relevant youtube videos on hacking and computer ethics

Interdisciplinary Connections

MA.K-12.4	Model with mathematics.
MA.K-12.8	Look for and express regularity in repeated reasoning.
LA.RST.11-12.3	Follow precisely a complex multistep procedure when carrying out experiments, taking measurements, or performing technical tasks; analyze the specific results based on explanations in the text.
LA.RST.11-12.4	Determine the meaning of symbols, key terms, and other domain-specific words and phrases as they are used in a specific scientific or technical context relevant to grades 11-12 texts and topics.
LA.RST.11-12.9	Synthesize information from a range of sources (e.g., texts, experiments, simulations) into a coherent understanding of a process, phenomenon, or concept, resolving conflicting information when possible.