South Central College

COMP 1130 Programming Fundamentals

Common Course Outline

Course Information

Description Programming Fundamentals teaches you how to design and develop small programs that solve different problems and implement ideas. In this class, you will discover how applications store and work with data, make decisions, and perform repetitive tasks. You will use and create functions, data structures, and objects, to represent the concepts from real life in your code. You will also use programming language libraries to develop efficient code that is easy to maintain. In the process of learning, we will emphasize testing, debugging, and scalability of your programs. (Prerequisites: COMP1120 Foundations of Computing)

Total Credits 4

Total Hours 64

Types of Instruction

Instruction Type	Credits/Hours
Lecture	4/64

Pre/Corequisites

Prerequisite COMP1120 Foundations of Computing

Institutional Core Competencies

Critical and Creative Thinking - Students will be able to demonstrate purposeful thinking with the goal of using a creative process for developing and building upon ideas and/or the goal of using a critical process for the analyzing and evaluating of ideas.

Course Competencies

1. Establish a working system for developing code.

Learning Objectives

Write programs that are well-formed and well documented.
Use an integrated development environment (IDE) to create, execute, test, and debug secure programs.

2. Use proper language syntax in expressions and statements.

Learning Objectives

Classify the elements of the programming language's syntax.
Use the assignment operator.
Use arithmetic operators.
Utilize relational operators.

Utilize logical operators.

3. Utilize data types.

Learning Objectives

Compare and contrast the primitive data types of a programming language.
Describe how each primitive data type is stored in memory.
Identify the criteria for data type selection.
Use variables to store and work with data.
Explain the scope of variables.

4. Implement control structures and logic in programming.

Learning Objectives

Employ decision-making in your programs using if/else statements.
Construct repetitive tasks using loops.
Nest if/else statements and loops.
Analyze control structures, exceptions, and regular expressions.

5. Use and create functions.

Learning Objectives

Define built-in functions.
Declare and call custom functions.
Write a function that is expecting parameters.
Call a function passing parameters to it.
Call a function that returns data.
Decompose a program into subtasks and use parameter passing to exchange information between the subparts.

6. Use data structures.

Learning Objectives

Explore data structures by building arrays.
Manipulate arrays with built-in methods.
Utilize arrays to store data.

7. Apply basic object-oriented programming concepts.

Learning Objectives

Create objects from predefined classes.
Manipulate numbers, strings, and dates by using standard libraries for a given programming language.
Explain the concepts of an object’s properties and methods.
Create custom objects.

8. Incorporate events and event handling for user participation.

Learning Objectives

Define concepts of an event listener and an event handler.
Identify common events in a program.
Incorporate events in your applications.

9. Produce graphical user interfaces that incorporate simple color models and handle events.

Learning Objectives

Implement a graphical user interface in a program.
Demonstrate events for the graphical user interface.
Incorporate data validation into programs.
Illustrate how to maintain state in a program.

10. Apply a variety of strategies to test and debug programs.

Learning Objectives

Use tools to monitor program flow.
Verify program correctness through the development of sound test plans and the implementation of comprehensive test cases.
Step through a program using debugging tools.

Identify different types of testing, including security, unit testing, system testing, integration testing, and interface usability.

11. Utilize libraries to save time.

Learning Objectives

Compare external libraries for ease of use, power, documentation.
Explain the programming library concept.
Use standard libraries for a given programming language.
Investigate the risks in using third-party applications, software tools, and libraries.

12. Use strategies to work with other systems.

Learning Objectives

Utilize application programming interfaces.
Use external data in your programs.

13. Apply the program development process to problems that are solved using fundamental programming constructs and predefined data structures.

Learning Objectives

Describe input/output programming.
Establish the steps in designing and implementing a program.
Diagram the phases of the secure software development lifecycle (SecSDLC).
Illustrate common behaviors that contribute to the effective functioning of a team.
Create modularized programs.
Use standard analysis and design techniques to produce a team-developed, medium-sized, secure software application that is fully implemented and formally tested.