Miami Dade College

Course Description

COP1334 | Introduction to C++ Programming | 4.00 credits

This course is designed for students in technology majors who require a foundation in computer programming. Students will learn the syntax and rules of the C++ language, including how to code, compile, debug and execute programs. Students will learn program design, structured and modular programming, arrays, and file processing. No previous computer courses are required although CGS 1060C is recommended.

Course Competencies

Competency 1: the student will demonstrate an understanding of the program development process by:

1. Writing pseudocode for program development before writing the code.
2. Applying the techniques of functional decomposition to break a programming design problem into smaller pieces.
3. Incorporating adequate and meaningful comments into the source code of programming projects.
4. Participating in a team to develop a solution to a problem.
5. Testing and debugging programming logic and code.

Learning Outcomes

1. Communication
2. Numbers and Data
3. Critical Thinking
4. Computer/Technology Usage

Competency 2: the student will demonstrate an understanding of mastery of basic C++ fundamental data types and operators by:

1. Using all the data types (float points, integers, long, double, boolean, characters, and strings) available in C++ for programming assignments.
2. Using descriptive and meaningful names in programming assignments.
3. Creating programs that use casting of data types.
4. Creating programs that use all existing operator(s) (+,-,*, %,/,=) available in C++.
5. Explaining the properties of a variable, such as its name, value, scope, persistence, and size.

Competency 3: the student will demonstrate an understanding of conditional statements by:

1. Creating programs that use if, else, if, and else statements to evaluate conditions.
2. Creating a program that uses logical operators (and, not, or) in conditional statements.
3. Creating a program that uses comparison operators (==, <, >, <=, >=) in conditional statements.
4. Creating a program that uses the switch, case, and break conditional structure to evaluate the conditions.
5. Creating a program that uses nested conditional statements.

Competency 4: the student will demonstrate an understanding of loops by:

1. Creating programs that use while, do-while, and loops to create repetition.
2. Analyzing existing programs with loops and determining the results.
3. Creating programs that use nested loops.

Competency 5: the student will demonstrate an understanding of mastery of functions by:

1. Creating functions that use call-by-reference and call-by-value.
2. Modifying existing programs that use functions.
3. Creating programs that include and use existing C++ library functions.
4. Creating a program that uses functions to return values.
5. Identifying the scope of variables in functions.

Competency 6: the student will demonstrate an understanding of structured data types by:

1. Explaining the form and uses of the array.
2. Creating a program that uses single and multi-dimensioned arrays.
3. Evaluating existing programs that search arrays.
4. Writing a program that uses structures.
5. Explaining the concepts of classes, data abstraction, encapsulation, and polymorphism.

Competency 7: the student will demonstrate an understanding of the input and output functions of a program by:

1. Writing a program that reads an existing sequential file.
2. Writing a program that creates a sequential file.
3. Writing a program that produces formatted printed output.
4. Modifying a program that produces formatted printed output.