**NORTHWESTERN CONNECTICUT COMMUNITY COLLEGE**

**COURSE SYLLABUS**

**Course Title:** Object Oriented Programming Using C++ **Course #:** CSC* 213

**Course Description**: 3 Credits. Introduces students to the C++ programming language. Techniques for solving problems with both numerical and non-numerical applications will be explored, incorporating rules of syntax, expressions and operators. Sequential and direct-access file processing are discussed. Concepts and examples of data types, recursive & virtual functions, arrays, pointers, vectors, strings, namespaces, data abstraction with classes, objects, overloading, inheritance, and data structures are presented.
.

**Pre-requisite** CSC* 104

**Goals:** Students are expected to

- discuss the history of the C++ programming language
- construct executable and reusable program modules
- apply various programming paradigms
- apply algorithm design and development techniques
- apply sequence, selection and iteration control structures to program development
- apply standard debugging tools to problem solutions
- construct program code using problem-solving techniques and tools
- create specifications for testing and debugging
- construct program solutions using various data forms and basic data structures
- produce technical documentation

**Outcomes:** Upon successful completion of this course students will be able to:

1) Demonstrate an understanding of computing basics
   a) Describe computer basics and operating systems
   b) Explain number representation in multiple formats

2) Demonstrate the basic steps in developing a C++ program
   a) Explain the basic syntax of a C++ Program
   b) Write, run and save a simple C++ project
   c) Demonstrate performing calculations
   d) Demonstrate the use of variables, constants, methods and classes
   e) Distinguish syntax errors, runtime errors, and logic errors
   f) Explain correct programming style and naming conventions
   g) Explain the correct usage of various data types
   h) Identify the correct usage for Boolean data types
   i) Apply relational and logic operators to write Boolean expressions
   j) Demonstrate the use of Boolean expressions in control statements
   k) Implement select control
   l) Write expressions using the conditional operator
   m) Display formatted output

3) Demonstrate the use of iterative operations
   a) Use While, and For loops to control repetition of statements
   b) Explain the control of flow in loop statements

c) Use Boolean expressions to control loop statements
d) Explain the differences between different types of loops
e) Write nested loops
f) Demonstrate techniques for minimizing numerical errors
g) Implement program control with Break and Continue

4) Demonstrate the creation and use of methods
   a) Define and invoke methods
   b) Develop modular code that is easy to read, debug and maintain
   c) Determine the scope of variables
   d) Solve mathematical problems using methods
   e) Apply the concept of method abstraction to software development

5) Demonstrate the use of arrays in processing data
   a) Describe why arrays are necessary in programming
   b) Describe the steps in using arrays
   c) Create arrays and declare array reference variables
   d) Initialize the variables in an array
   e) Manipulate array contents
   f) Develop and invoke methods with array arguments
   g) Implement sorting and searching algorithms

6) Demonstrate an understanding of using objects
   a) Describe objects and classes, and use classes to model objects
   b) Construct objects using constructors
   c) Access objects using object reference variables
   d) Define a reference variable using s reference type
   e) Access an object's data and methods
   f) Declare a class and create an object from a class
   g) Distinguish between instance and static variables and methods
   h) Encapsulate data fields
   i) Develop methods with object arguments
   j) Distinguish among the String classes
   k) Explain how to pass command line arguments
   l) Explain the differences between procedural and object-oriented paradigms
   m) Design programs using object-oriented paradigms

7) Demonstrate the manipulation of classes
   a) Illustrate inheritance
   b) Distinguish between overriding and overloading
   c) Explain polymorphism, dynamic binding, and generic programming
   d) Prevent class extending and method overriding
   e) Design and use abstract classes

8) Demonstrate an understanding of software development
   a) Describe the software life cycle
   b) Determine responsibilities for a class
   c) Describe relationship types
   d) Declare classes that represent relationships