



## Course Syllabus

### **COSC 1436 – Programming Fundamentals I**

Catalog Description: This course introduces the fundamental concepts of structured programming. Topics include software development methodology, data types, control structures, functions, arrays, and the mechanics of running, testing, and debugging.

**Lecture hours = 4, Lab hours = 3**

**Prerequisites:** Computer Literacy

**Semester Credit Hours:** 4

**Lecture Hours per Week:** 3

**Contact Hours per Semester:** 96

**State Approval Code:** 11.0201.5507

**Course Subject/Catalog Number:** COSC 1436

**Course Title:** Programming Fundamentals I

#### **Instructional Goals and Purposes:**

The purpose of this course is to teach students the fundamentals of computer programming using the C++ language from a game programming perspective.

#### **Learning Objectives:**

1. Use basic elements of C++.
2. Understand truth and branching.
3. Understand for loops, strings and arrays.
4. Understand the standard template library
5. Create programs using functions.
6. Use references.
7. Understand the use of pointers.
8. Create programs using classes
9. Use advanced classes.
10. Code programs using dynamic memory.
11. Understand inheritance
12. Develop programs using polymorphism.

#### **Specific Course Objectives (includes SCANS):**

After studying the material presented in the text and online, the student should be able to complete all behavioral/learning objectives listed below with a minimum competency of 70% on assignments and exams.

1. **Use basic elements of C++.** (1a-i, 1a-iv, 1b-iv, 1b-vi, 1c-iv, 2c-i, 2c-ii, 2c-iii, 2c-iv)
  - a. Create input and output statements
  - b. Code arithmetic computations
  - c. Use variables
  - d. Use constants and enumerations
2. **Understand truth and branching.** (1a-i, 1a-ii, 1a-iv, 1b-iv, 1b-v, 2c-i, 2c-ii, 2c-iii, 2c-iv)
  - a. Code simple and compound conditions,
  - b. Use if statements, switch statements

- c. Implement do loops, and while loops
  - d. Understand the game loop
  - e. Generate random numbers
3. **Understand for loops, strings and arrays.** (1a-i, 1a-iv, 1b-iv, 2c-ii, 2c-iv)
    - a. Code statements iterating over sequences
    - b. Understand objects - data members and member functions
    - c. Use string objects and their member functions
    - d. Write modules using arrays
  4. **Understand the standard template library.** (1a-i, 1a-iv, 1b-i, 1b-iii, 1b-iv, 1b-vi, 2c-i, 2c-ii, 2c-iii, 2c-iv, 2d-iii)
    - a. Write modules using libraries
    - b. Understand vectors and other collections
    - c. Demonstrate the use of iterators and indirect access
    - d. Develop algorithms
    - e. Plan and design program logic with pseudocode
  5. **Create programs using functions.** (1a-i, 1a-ii, 1a-iv, 1b-iii, 1b-vi, 2c-i, 2c-ii, 2c-iii, 2c-iv)
    - a. Create modules using arguments and parameters
    - b. Write code using return values
    - c. Create global variables and global constants
    - d. Understand overloading functions
    - e. Explain the use of inlining functions
  6. **Use references.** (1a-i, 1a-iv, 1b-i, 1b-iii, 1b-iv, 1b-vi, 2c-i, 2c-ii, 2c-iii, 2c-iv, 2d-iii)
    - a. Write code referencing variables
    - b. Create modules passing references
    - c. Alter argument variables
    - d. Return references
  7. **Understand the use of pointers.** (1a-i, 1a-iv, 1b-iv, 1b-vi, 2c-i, 2c-ii, 2c-iii, 2c-iv)
    - a. Declare and initialize pointers
    - b. Dereferencing pointers
    - c. Create constants and pointers
    - d. Pass and return pointers
    - e. Create a module using pointers and arrays
  8. **Create programs using classes.** (1a-i, 1a-iii, 1a-iv, 1b-iv, 1b-vi, 2c-i, 2c-ii, 2c-iii, 2c-iv)
    - a. Create new types
    - b. Define data members and member functions
    - c. Instantiate objects
    - d. Develop constructors
  9. **Use advanced classes.** (1a-i, 1a-iv, 1b-iv, 1b-vi, 2c-i, 2c-ii, 2c-iii, 2c-iv)
    - a. Set member access levels
    - b. Create accessor member functions
    - c. Use constant member functions
    - d. Create modules using static data members and member functions
  10. **Code programs using dynamic memory.** (1a-i, 1a-iii, 1a-iv, 1b-iv, 1b-vi, 2c-i, 2c-ii, 2c-iii, 2c-iv).
    - a. Dynamically allocate memory
    - b. Acquire and free memory
    - c. Prevent memory leaks
    - d. Understand shallow and deep copies
  11. **Understand inheritance.** (1a-i, 1a-iv, 1b-iv, 1b-vi, 2c-i, 2c-ii, 2c-iii, 2c-iv)
    - a. Create base and derived classes
    - b. Use inherited data members and member functions
    - c. Override base class member functions
  12. **Develop programs using polymorphism.** (1a-i, 1a-iii, 1a-iv, 1b-iv, 1b-vi, 2c-i, 2c-ii, 2c-iii, 2c-iv)
    - a. Create virtual functions and polymorphic behavior
    - b. Develop pPure virtual functions and abstract classes
    - c. Split up code into multiple files

**Course Content:**

Students in all sections of Programming Fundamentals I will be required to do the following:

1. Students will submit computer programs for each learning module of the course. Each program must demonstrate comprehensive knowledge of the learning module represented.
2. Students will assemble a portfolio of programming projects to be submitted at the end of the semester.
3. Students will conduct an interactive presentation of a comprehensive programming project subject to peer and instructor evaluation.

**Methods of Instruction/Course Format/Delivery:**

Students in both the traditional class and in the Internet class will have access to this course via WebCT. Students in the traditional class will meet regularly for lecture over the material. Students in the Internet class will only be required to meet with the instructor for testing; however, Internet students are always welcome to attend the traditional class (especially for exam reviews). Resources provided through WebCT include

- A calendar displaying assignments each week (please check often)
- Online assignments
- Chapter notes
- Email (totally contained within WebCT)

All assignments will be submitted through WebCT. After the assignment has been graded, the student will be able to view his or her grade by returning to the assignment and clicking the View Scores button or by clicking the My Grades link in the left banner. All exams will be hands-on application tests and students will not be able to view the answers to the exams online; however, they will be able to see their grade in My Grades and drop by the office to review their exams. I generally will have your work graded and posted within two days following the deadline.

Students in both the traditional and Internet classes should use the Email within WebCT to communicate with the instructor. Using WebCT email gives you access to the instructor and other classmates without having to remember or type email addresses—you just select a name from the list. If you are not able to contact me using email in WebCT, you may use my Panola College email address. I attempt to respond to all email within 24 hours. If you make an appointment with me through email to take an exam, for example, I will reply to your email—if I do not reply you should send your email to me again or call me. Please always include a subject line and your name in your email.

**Assessment:**

The following items will be assigned during the semester and used to calculate the student's final grade:

- **ASSIGNMENTS**  
We will work through each of the learning modules which correspond to the chapters in your textbook. At the end of each learning module, you will complete a programming project demonstrating your knowledge of the programming concepts presented in the learning module. Program source code will be submitted to me according to the schedule provided using the online drop box in the Assignments link of WebCT.
- **PORTFOLIO**  
The portfolio will be a collection of all program source code developed during the semester including a comprehensive programming project to be completed in the final weeks of the semester.  
  
Portfolios are due by the scheduled deadline.
- **EXAMS**  
There will be one assessment to verify that you have the comprehensive knowledge required to produce your portfolio. You will demonstrate this knowledge by conducting an interactive presentation of a comprehensive programming project subject to peer and instructor evaluation.

**Course Grade:**

The grading scale for this course is as follows:

- Assignments – 20%
- Portfolio – 50%
- Exams – 30%

All of your grades including a mid-semester and final grade will be posted to My Grades in WebCT.

**Texts, Materials, and Supplies:**

- *Beginning C++ Game Programming*, Michael Dawson, Course Technology, 2004, ISBN: 1-59200-205-6.
- Access to a computer and the Internet.
- Student data files and Dev-C++ are provided with the textbook.

**Other:**

- For current texts and materials, use the following link to access bookstore listings: <http://www.panola.edu/collegestore.htm>
- For testing services, use the following link: <http://www.panola.edu/instruction/dl/testing.htm>