

LAST REVIEW: 2010-2011

NEXT REVIEW: 2015-2016

STATUS: A

COURSE TITLE: Intermediate C++ Programming

COMMON COURSE NUMBER: COP 1335C

CREDIT HOURS: 3

CONTACT HOUR BREAKDOWN

(per 16 week term)

CLOCK HOURS:

(Voc. Course ONLY)

Lecture: 48

Lab: 16

Clinic:

Other: 0

PREREQUISITE(S): CIS 1000C and COP 1334C (each with a grade of C or higher)

COREQUISITE(S):

PRE/COREQUISITE(S):

COURSE DESCRIPTION *(750 characters, maximum):*

Upon successful completion of this course, the students should be able to design and code robust, structured, well-documented C++ programs to solve complex problems.

General Education Requirements – Associate of Arts Degree (AA), meets Area(s):

Area

General Education Requirements – Associate in Science Degree (AS), meets Area(s):

Area

General Education Requirements – Associate in Applied Science Degree (AAS), meets Area(s):

Area

UNIT TITLES

- 1. Arrays, Structs and Their Applications (review)**
- 2. Pointers**
- 3. Classes and Data Abstraction**
- 4. Friend Functions and Classes**
- 5. Inheritance**
- 6. Polymorphism**
- 7. Virtual Functions**
- 8. Streams**
- 9. Templates**
- 10. Exception Handling**

EVALUATION:

Please provide a brief description (250 characters maximum) that details how students will be evaluated on the course outcomes.

Evaluation instruments will include written and/or skills-based examination and individual in-class and/or take-home assignments. Evaluation methods may also include group in-class and/or take-home assignments.

****Required Competencies****1) Read with critical comprehension.**

The student will be introduced to the basic texts, concepts, vocabulary, and methods necessary for developing an understanding of the discipline and meeting the required benchmarks as stated in the course outline.

2) Write clearly and coherently.

The student will demonstrate an understanding and mastery of subject matter in a variety of ways, including writing. Writing activities may include both graded and ungraded essays, short answer quizzes, summaries, reactions, journals, and various other reports.

3) Demonstrate and apply literacy across all the disciplines (indicate which ones apply).

- a) **Information literacy** means understanding how to locate needed information, using the appropriate technology for the task, managing and evaluating the extracted information and using it effectively and ethically.
- b) **Technology literacy** is the ability to responsibly and effectively use appropriate technology to access, manage, integrate, or create information, and/or use technology to accomplish a given task.
- c) **Workplace literacy** is having the appropriate knowledge and skills to communicate and work with others effectively and perform job duties, whether it is through the use of computers and/or other technology.
- d) **Cultural literacy** is recognizing, understanding, and appreciating the similarities and differences between one's own culture and the cultures of others through a study of the arts, customs, beliefs, values, and history that define a culture.
- e) **Quantitative literacy** is having the ability to formulate, solve and interpret mathematical/statistical operations and graphical/tabular representations to make informed decisions.
- f) **Scientific literacy** means understanding the methodology and application of the scientific process, the physical and biological worlds, and recognizing that scientific knowledge is continuously updated or revised as new information is discovered.

4. Apply problem-solving skills or methods to make informed decisions in a variety of contexts.

The student will use acquired skills or methods to recognize, analyze, adapt, and apply critical thinking to solve problems and make informed decisions.

EVALUATION:

In the box to the right of the Methods of Assessment, enter all specific learning outcome numbers (i.e. 1.1, 2.7, 4.0, 4.2 and 5.12) that apply.

1. Portfolio	
2. Short essays	
3. Research Papers	
4. Group projects	
5. Discussions (In class and online)	
6. Multiple Choice tests	
7. Presentations	
8. Service Learning Projects	
9. Pop quizzes	
10. Take-home tests	
11. Summaries and critiques	
12. Reaction papers	
13. Surveys	
14. Performance	
15. Short answer tests	
16. Classroom debates and colloquia	
17. Blogs, wikis, web pages	
18. Other (Please explain)	

UNITS

Unit 1 Arrays, Structs, and Their Applications (review)

General Outcome:

1.0 The student shall:

Be able to use more complex applications of C++ structured data types.

Specific Measurable Learning Outcomes:

Upon successful completion of this unit, the student shall be able to:

1.1 Discuss the difference between simple and structured C++ data types

1.2 Describe various applications of arrays

1.3 Define and use one-dimensional arrays

1.4 Discuss and use sorting algorithms to sort an array

1.5 Discuss and use searching strategies to search an array

1.6 Define and use parallel arrays

1.7 Define and use multidimensional arrays

1.8 Discuss the various applications of records (C++ structs)

1.9 Define and use structs and arrays of structs

Unit 2 Pointers

General Outcome:

2.0 The student shall:

Be able to create links between multiple structures; and to facilitate the allocation and de-allocation of memory.

Specific Measurable Learning Outcomes:

Upon successful completion of this unit, the student shall be able to:

2.1 Explain the use of pointers

2.2 Declare and use pointers

2.3 Use pointers to access memory addresses

2.4 Use pointers to link structures together

2.5 Explain and use pointer arithmetic

2.6 Explain the relationship between pointers and arrays

Unit 3 Classes and Data Abstraction

General Outcome:

3.0 The student shall:

Be able to define classes and access public, private, and protected member data items and methods (functions).

Specific Measurable Learning Outcomes:

Upon successful completion of this unit, the student shall be able to:

3.1 Explain the concept of data abstraction

3.2 Explain the concept of classes

3.3 Use classes as a means of implementing data abstraction

3.4 Discuss class scope and the accessing of class members

3.5 Explain and use public, protected and private class members

3.6 Describe and use constructors (including default arguments)

3.7 Describe and use destructors

3.8 Define and use hierarchical classes

Unit 4 Friend Functions and Classes

General Outcome:

4.0 The student shall:

Be able to use friend classes and functions to facilitate communication with class members.

Specific Measurable Learning Outcomes:

Upon successful completion of this unit, the student shall be able to:

4.1 Discuss the concept of friend functions

4.2 Define and use friend functions to access class member data

4.3 Discuss the concept of friend classes

4.4 Define and use friend classes as bridges between classes

Unit 5 Inheritance

General Outcome:

5.0 The student shall:

Be able to develop base classes and then derive concrete classes from them.

Specific Measurable Learning Outcomes:

Upon successful completion of this unit, the student shall be able to:

5.1 Explain the concept of inheritance

5.2 Discuss the use of inheritance in program design

5.3 Create base classes

5.4 Derive classes from a base class

5.5 Redefine base class members in a derived class

5.6 Discuss and use public, protected and private base classes

5.7 Discuss and use multiple inheritance

Unit 6 Polymorphism

General Outcome:

6.0 The student shall:

Be able to explore the concept and practice of polymorphism.

Specific Measurable Learning Outcomes:

Upon successful completion of this unit, the student shall be able to:

6.1 Explain the concept of polymorphism

6.2 Discuss and use operator overloading

6.3 Discuss and use function overloading

Unit 7 Virtual Functions

General Outcome:

7.0 The student shall:

Be able to write virtual functions to create abstract base classes from they can derive new classes.

Specific Measurable Learning Outcomes:

Upon successful completion of this unit, the student shall be able to:

7.1 Discuss the concept of (and implement) virtual functions

7.2 Create abstract base classes through the use of virtual functions

7.3 Use virtual functions to implement polymorphism

7.4 Explain and use virtual destructors

Unit 8 Streams

General Outcome:

8.0 The student shall:

Be able to demonstrate C++'s input and output streams.

Specific Measurable Learning Outcomes:

Upon successful completion of this unit, the student shall be able to:

8.1 Discuss and use iostream header files

8.2 Discuss and use stream input and output

8.3 Discuss and use stream manipulators

8.4 Discuss and use stream format states

8.5 Explain and perform unformatted I/O

8.6 Create, read from and update a sequential access file

8.7 Create, read from and write to a random access file

8.8 Input and output objects

Unit 9 Templates

General Outcome:

9.0 The student shall:

Be able to use templates in program design and development.

Specific Measurable Learning Outcomes:

Upon successful completion of this unit, the student shall be able to:

9.1 Discuss and use function templates

9.2 Overload function templates

9.3 Discuss and use class templates

9.4 Explain class templates and non-type parameters

9.5 Discuss the relationship between templates and inheritance

9.6 Discuss the use of friends with templates

9.7 Explain and use static data members with templates

Unit 10 Exception Handling

General Outcome:

10.0 The student shall:

Be able to incorporate C++'s exception handling features into their programs.

Specific Measurable Learning Outcomes:

Upon successful completion of this unit, the student shall be able to:

10.1 Discuss the concept of exception handling

10.2 Explain when exception handling should be used

10.3 Describe and use try and catch blocks

10.4 Throw an exception

10.5 Catch an exception

10.6 Rethrow an exception

10.7 Throw a conditional expression

10.8 Discuss constructors, destructors and exception handling

10.9 Explain the relationship between exceptions and inheritance.