Administrative Unit: Computer and Mathematical Sciences Department

Course Prefix and Number: CISS 245

Course Title: Advanced Programming

Number of: Credit Hours: 4  Lecture Hours: 3  Laboratory Hours: 1

Catalog Description: A continuation of CISS 240. Topics include object-oriented programming in C++ with classes, methods, inheritance, dynamic memory allocation, file streams, linked lists, and stacks. Prerequisite: Grade of C or better in CISS 240.

Prerequisite(s)/Corequisite(s): Grade of C or better in CISS 240.

Text(s): Most current edition of the following:

Dietel, H. M. & Dietel, P. J. C++ How to Program. Prentice Hall.

Course Objectives:

• To understand the object-oriented paradigm and practice of object-oriented programming.
• To understand input and output streams.
• To understand and implement basic data structures such as stacks and linked lists.

Measurable Learning Outcomes: Questions (Q) and exercises (E) are provided to demonstrate the basic level of competence expected of the student.

• Understand and implement a class.
• Understand use and define constructors with or without initializer lists.
Q: What is a copy constructor and how would it be used?
E: Demonstrate how a constructor can provide default values and show how to invoke such a constructor.
Q: Is it possible to have a constant data member in a class? If you answer yes, demonstrate how such a data member would be initialized.
• Understand and create destructors.
Q: When should a default destructor not be used?
• Understand and explain the difference between public and private members.
• Understand, use and write get and set methods.
Q: What methods might be appropriate for an employee class that recorded basic information for a corporation's employees?
Q: What are set and get methods?
• Understand, use and write static members.
Q: Is it possible to access a static class data member even though not objects of the class have been
• Understand, use and write constant methods.
  Q: What does it mean when a class method is declared as constant?
  Q: In general, which class methods should be declared constant and why?
• Understand the use of a pointer in a method.
  E: Demonstrate how you would use this pointer to print the private data members of a class.
• Understand and provide class support for constant objects.
• Understand and write friend functions.
• Understand and write friend classes.
• Understand and write overloaded operators.
  Q: Is it possible to overload both the pre- and post-increment operators? If you answer yes, explain how the compiler is able to distinguish the function signatures since the operator is the same in both cases.
  Q: Can brackets, ( ), be overloaded?
  Q: Can parentheses, ( ), be overloaded?
  Q: Under what circumstances must an overloaded operator be declared as a friend class?
  Q: For the questions below assume the existence of a simple Name class that stores first and last names as character strings using dynamically acquired memory. Private data members will consist of two pointers, one to the first name, firstNamePtr. and one to the last name, lastNamePtr.
  • Show how to overload the stream insertion operator to print a Name object.
  • Write a destructor for the class.
  • Can the default assignment operator be used for this class or is it necessary to explicitly overload the assignment operator?
  • Under what circumstances is it necessary for the programmer to explicitly code a destructor?
  • Given objects N1 and N2 of the Name class, write one or more class methods that would support the following syntax:
    if (N1 < N2 ) …
    if (N1 < “Marry Sue”) …
    if (“Marry Sue” < N1) …
  • Understand and write subclasses via public, protected and private inheritance.
  Q: What is the difference between public and protected inheritance?
  Q: Assume the following classes:
  class Klondar
  {
    public:
      klondar ( int = 0 );
      void kMethod1 (int) const;
      void kMethod2 (int) const;
protected:
    int kInt1;
};

class foo: private klondar
{

    public:
    foo (int = 0);
    void kMethod1 (int) const;
    void kMethod2 (int) const;

    private:
    int fInt1;
};

• If a third class called foobar does public inheritance from foo, what klondar members, if any, can be accessed from foobar objects? What foo members, if any, can be accessed from foobar objects? What foo members, if any, can be accessed from foobar objects?
• What klondar members, if any, can be accessed from a foo object?
• If foo did public rather than private inheritance from klondar, would your answer to the above question change? Explain.
• Assuming the following declaration,
    foo f1;
    foo *fooPtr = &f1;
    demonstrate how you could use fooPtr to access all data and function members that are accessible via fooPtr.

• Understand and write virtual functions and pure virtual functions.
• Understand and use polymorphism.
  Q: What is a virtual function?
  Q: What is pure virtual function? How are pure virtual functions declared?
  Q: What is polymorphism?
  Q: Is it necessary to override a base class virtual function in a subclass? Explain. Would your answer change if the base class function were pure? Explain.
• Understand and use class composition.
• Open a file object for read, write or append.
  E: Write a program that will read a series of names entered from the keyboard by a user and store the names in a sequential file, in a random file.
• Implement a template stack class.
  E. Write the definition for a simple stack template class that supports pop and push operations. The implementation should use an array with a default size of 10 for storing the stack. You should also provide an integer that keeps track of how many items are currently stored on a stack. You should also provide isEmpty and isFull Boolean methods that return true/false depending on whether the stack is empty or full. Show how to declare a stack for holding
characters and stack for holding doubles.
- Implement a template linked list class.
- E: Create a linked list template class that supports the following methods: reverseList, mergeList, and an overloaded assignment (=) operator.
- Implement a class to solve problems.
  E: Implement a directory class. The class should provide basic storage and lookup capabilities.

Topical Outline (major areas of coverage):
- Structure and structure definitions
- Classes and class definitions
- Constructors and destructors
- Constant objects and member functions
- Static class members
- Dynamic memory management in class objects
- Operator overloading
- Class inheritance
- Polymorphism and virtual functions
- Class templates
- Stream input/output
- Linked lists
- Stacks

Recommended maximum class size for this course: 20

Library Resources: Online databases are available at http://www.ccis.edu/offices/library/resources.asp. You may access them from off-campus using your eServices login and password when prompted.

Prepared by: Yihsiang Liow

Date: February 1, 2005

NOTE: The intention of the master syllabus is to provide an outline of the contents of this course, as specified by the faculty of Columbia College, regardless of who teaches the course, when it is taught or where it is taught. Faculty members teaching this course for Columbia College are expected to facilitate learning pursuant to the course objectives and cover the subjects listed in the topical outline. However, instructors are also encouraged to cover additional topics of interest so long as those topics are relevant to the course’s subject. The master syllabus is, therefore, prescriptive in nature but also allows for a diversity of individual approaches to course material.

Office of Academic Affairs
12/04