Course Syllabus for CSE 283, Spring 2017

COURSE INFORMATION

Title: Introduction to Object-Oriented Design

Instructor: C-Y (Roger) Chen **Email:** crchen@syr.edu

Office Location: CST 4-281 **Office Hours:** TBA; others by email appointment

TA: Chunxu Tang; **Office Hours:** TBA

COURSE DESCRIPTION

Presents fundamental software design concepts of functional decomposition and object-oriented design. Uses C++ to implement design projects that will be completed to demonstrate the design concepts. If time permits, we might spend final 2-3 weeks covering Java.

COURSE OBJECTIVES

The students will…

- Review the classical software design concepts of functional decomposition.
- Subsequently learn object-oriented design concepts.
- Be able to design from a functional decomposition perspective.
- Be able to develop similar object oriented designs in C++.
- Be able to compare and contrast different design methodologies and languages.

The above maps to three Accreditation Board for Engineering and Technology (ABET) educational objectives for computer engineering courses:

The educational objective of the Bachelor of Science in Computer Engineering (BSCE) program in the Department of Electrical Engineering and Computer Science (EECS) at Syracuse University is to prepare well-rounded graduates that are ready for work and ready for change.
I. Well-rounded graduates of the BSCE program are known by their professional competence, innovative thinking, willingness to further enhance their education, ability to work individually and in diverse teams, leadership abilities, communication skills, and integrity.
II. Graduates of the BSCE program who are ready for work are engaged in applying the knowledge acquired in Computer Engineering, combined with their problem solving abilities, to produce feasible solutions to problems, in a timely manner, which are deemed important in industry, government, or academia.
III. Graduates of the BSCE program who are ready for change exhibit the intellectual

flexibility necessary to solve new problems in innovative ways by integrating multiple viewpoints from several disciplines in search of the best possible solutions or applying their knowledge to different professional disciplines.

READING MATERIAL

Textbooks:
(Required) C++: A Beginner's Guide, 2nd Edition
Author: Herbert Schildt

Primary sources of lectures: Material prepared by the instructor

GRADING & ASSESSMENT PROCESS

Homework: You are responsible for all 10-12 assignments.
(~55% of final grade)

Weekly Quizzes: Beginning of Wednesday classes (~15%)

Midterm Exam 2: (~10% of final grade)

Final Exam: There will be a comprehensive final on all chapters covered in the course.
(~20% of final grade)

Class Attendance and Participation: class attendance is strongly mandatory.
You are expected to participate in class activities and discussions. Class attendance may affect your grade.

Direct Assessment: The homework assignments and discussions during class, particularly the software design portions and programming in C++ and OOD, will be used to assess your ability to meet the ABET requirements for developing an understanding of abstraction and refinement of computing systems.

REQUIRED KNOWLEDGE, ABILITIES, AND SKILLS

Pre-Requisite: ESC 102 – Introduction to Computing (or other equivalent programming knowledge)

Recall: You should be able to…

- Define the Boolean operations NOT, AND, and OR.
- Define the precedence of mathematical operators.

Comprehension: You should be able to…

- When given a Boolean expression, express it using truth tables.
- When given a high-level programming statement, explain its meaning in English.
- When given an integer number in decimal, binary, or hexadecimal representation, translate it to another base.
- Add and subtract numbers represented in decimal, binary or hexadecimal.
- When given a number, form its two’s complement.

Synthesis: You should be able to…

- When given a problem description, develop a C programming solution.

TOPICS COVERED

- Functional Decomposition
- Reuse
- Detailed Mechanisms behind Function Invocation
- Object-Oriented Design
- Encapsulation
- Overloading and Defaults
- Arrays of Objects
- Pointers to Objects
- Linked list
- Manipulation of linked lists and their Application
- Left and Right References
- Operator Overloading
- Applications of operator and reference overloading
- I/O overloading
- Recursion
- Standard Template Library (STL) – vector, list, map, queue, stack, deque, set
- Iterator

- Manipulation and Applications of STL containers
- Inheritance
- Dynamic Allocation and Deallocation
- Memory Management
- Polymorphism
- Templates and Manipulators
- Exception Handling
- Multi-File Program development
- Move constructor, copy constructor, overloading assignment operator
- Optimization for big-data applications
- Stream for I/0
- Threaded Programming
- Mutex and various locking and notification mechanisms for synchronization
- Centralized locking vs distributed locking

ACQUIRED KNOWLEDGE, ABILITIES, AND SKILLS

Comprehension: You will be able to…

- Recognize what polymorphism can be used for and how it increases reuse potential.

- Describe the application and use of the following: Java classes, Java formal interface specifications, Java applets, Java modifiers, Java graphics, Java threads, and the Java virtual machine.

- Describe the application and use of the following: streams, function passing, dynamic allocation, friends, manipulators, overloading, virtual members and abstract classes, templates, and the Standard Template Library.

- When given a design problem, identify related data and functions to be encapsulated.

Analysis: You will be able to…

- When given a design problem, analyze the classes required to determine if inheritance can be efficiently utilized.

Synthesis: You will be able to…

- Develop code for object-oriented designs in C++.

- Implement constructors and destructors.

- Design classes to implement encapsulation in C++.

- Formulate strategies for data hiding using access rights and access methods.

- Develop tests to verify correct operation of designs.

- When given a design problem, develop a class/object model, dynamic model, and a functional model.

Evaluation: You will be able to…

- Compare and contrast the different design methodologies and languages.

- Evaluate an existing C++ implementation to determine what object-oriented design models are the bases of the program.

- Evaluate designs for reuse.

EDUCATIONAL OUTCOME TABLE

Computer Engineering Educational Outcomes[2] CSE 283 (*Bold: items related to CSE283)

- **a. an ability to apply knowledge of mathematics, science, and engineering**
- **b. an ability to design and conduct experiments, as well as to analyze and interpret data**
- c. an ability to design a system, component, or process to meet desired needs within realistic constraints such as economic, environmental, social, political, ethical, health and safety, manufacturability, and sustainability
- d. an ability to function on multidisciplinary teams
- e. an ability to identify, formulate, and solve engineering problems
- f. an understanding of professional and ethical responsibility
- **g. an ability to communicate effectively**
- h. the broad education necessary to understand the impact of engineering solutions in a global, economic, environmental, and societal context
- **i. a recognition of the need for, and an ability to engage in life-long learning**
- j. a knowledge of contemporary issues
- **k. an ability to use the techniques, skills, and modern engineering tools necessary for engineering practice.**
- **l. an ability to verify design correctness and evaluate performance of computing systems.**

CSE 283 Honor Policy

As stated in the course syllabus, every student is expected to behave ethically: do not cheat, plagiarize, or commit fraud.

Fraud includes manipulating the simulation results to make it appear that a design functions correctly when it does not; plagiarism includes using someone else's work without proper credit. The following guidelines further detail these definitions:

1. If you happen to find a solution to a problem in a written source (e.g., in a textbook or on the web) and use the solution, you should state the original source. It is unethical and plagiarism to use someone else's work without proper credit.
2. If you get help from someone (either a classmate or someone else) on a problem, you should state that (and give their name). Again, it is unethical to use someone else's ideas without giving proper credit. In both of the above cases, it is cheating and fraud to pass off something as being your work when it is not.
3. In this course, it is legitimate to discuss problems with each other at a conceptual level: for example, it's okay to figure out how to break a design problem up into smaller, easier-to-design modules, or to discuss general approaches to solving a problem. However, the final design solution must be your own: it is not legitimate to share any details, layouts, or other written solutions, or to discuss how to implement any design modules with anyone.
4. If you are unsure whether a certain action constitutes cheating, fraud, or plagiarism, assume that it does: you may ask me for clarification at any time.