CSCE 240 - Introduction to Software Engineering
Credit Hours: 3 hours
Contact Hours: 3 lecture hours
Instructors: Drs. Buell, Tang, Valafar

Required Textbooks: Walter Savitch, *Absolute C++, 4th or 5th edition*, Addison Wesley, 2009.
Bulletin Description: Fundamentals of software design and development; software implementation strategies; object-oriented design techniques; ethics in software development.
Prerequisites: CSCE 215, grade of C or better in CSCE 146

Required Course in CE, CIS, and CS programs
Learning Outcomes: Students will be able to:

1. Independently design and implement C++ programs in a Unix environment
2. Demonstrate mastery of pointers, iterators, memory management including object creation and destruction, and parameter passing in C++.
3. Demonstrate mastery of object oriented programming concepts including: inheritance, polymorphism, operator overloading, template functions and classes, and the use of STL containers.
4. Develop object oriented models using UML
5. Engage in program design and implementation in a team environment.
6. Use a source control tool in a team environment.

Student (Program) Outcomes addressed by course (Detailed mappings of these course outcomes to the Student Outcomes of the programs are in the detailed syllabus and the Assessment plan.)

Student Program Outcomes	SOs supported	SOs Moderately supported
Computer Engineering	a,b,c,e,k	f
Computer Information Systems	a, b, c, i, IS-j	e
Computer Science	a, b, c, i, CS-j, CS-k	e

Topics covered:

1. Unix Programming Environment: Unix tools, C preprocessor, Make, Shell, I/O redirection, debugging.
2. Pointers: Pointer manipulation, functions and function pointers, virtual functions.
3. Basic class management: constructors, destructors, data hiding, container classes.
4. Memory management: object creation and destruction, memory leak.
5. Advanced C++ features: operator overloading, iteration, special containers, inheritance, code reuse, multiple inheritance, virtual functions, polymorphism, templates, template libraries.
6. UML and object oriented modeling: use-case models, object identification, specifying static behavior, activity diagrams, collaboration diagrams and sequence diagrams, specifying relationships: generalization/specialization, aggregation, …
7. Introduction to Source Control and Distributed Source Control, for example, using git