Course Number: CSC 413
Course Title: Software Development
Number of Credits: 3
Schedule: Three hours of lecture/discussion per week.
Prerequisite: CSC 340 and CSC 412 with grades of C or better.

Catalog Description
Design and development of modern software applications. Object-oriented techniques: encapsulation, inheritance, and polymorphism as mechanisms for data design and problem solution. Software design, debugging, testing, and UI design. Software maintenance. Software development tools.

Expanded Description

Introduction to Software Development

Introduction to Object-Oriented Programming - OOP
Information Hiding, Class Hierarchy

The Java Language

Object Oriented Design
Plan for Change, Software Components, Interfaces vs. Implementation
Naming

A Comparision of Java and C++

A Compiler
Extended Example, Source, Tokens, AST, Decorated AST's, Code generation, Bytecodes

Lexical Analysis

Parsing - Syntax Analysis of the Token Stream Yielding the AST
Grammar for X, ASTS Built from Source Programs

Tree Visitors

Inheritance
Subclass, Subtypes and Substitutability, Forms of Inheritance, Modifiers
Benefits of Inheritance, Cost of Inheritance

The Interpreter
Frames (Activation Records)
Javadoc Documentation of Selected Interpreter Classes
The Runtime Stack, The Virtual Machine

Constraining (Decorating the AST; Type Checking)

Variable Scopes, Symbol Tables, Constraining Activities:

Code Generation
Frames (Activation Records), Runtime stack, Blocks

Mechanisms for Software Reuse
Inheritance vs. Composition (aggregation), Abstract classes vs. Interfaces, Combining Composition and Inheritance, Dynamic Composition

Implications of Inheritance
Polymorphic Variables, Memory Layout, Assignment, Clones (Shallow vs. Deep) Garbage Collection

Polymorphism
Polymorphic Variables, Overloading, Overriding, Replacement and Refinement Abstract Methods, Efficiency and Polymorphism

Input and Output Streams - Effective Uses of Inheritance with Composition
Readers, InputStreams

Exception Handling in Java

Collection Classes
Arrays, Lists, Properties, System Properties

Application Profiling
Used to tune performance

Course Objectives and Role in Program
The objectives of this course include:

- Teach important object oriented programming principles using a large application as a vehicle for learning; consider issues of *programming in the large*
- Introduce the student to integrated development environments.
- Teach the Java programming language, along with well-utilized Java library resources
- Expose the student to other software development tools including debuggers and code profilers

Students will develop several small applications and at least one large application.
The knowledge of software development tools and object oriented programming plays an important role in all software development projects students develop for courses in the program.

Learning Outcomes
At the end of this course students will

- Be able to write Java programs utilizing an integrated development environment
- Utilize a debugger when doing software development
- Apply object oriented programming principles effectively when developing small to medium sized projects

- Write robust code utilizing exception handling language features
- Use a code profiler to tune a program's performance

Method of Evaluation

Student learning will be evaluated on the basis of

- Completeness and quality of programming assignments.
- Grade on midterm examination
- Grade on final examination
- Class participation.

The weight assigned to each element of evaluation will be determined by the instructor of the course on the first day of the class.

Required Textbooks

Understanding Object-Oriented Programming with Java, Budd, T., Addison-Wesley, 2000

Core Java(TM) 2, Volume I--Fundamentals, Horstmann, C.S. and Cornell, G. Prentice-Hall

Modified by: B. Levine

Last Revision Approved: August 31, 2016