Course Syllabus

Computer Science

CSC142

Bob Driggs
Dean

Cate Sheller
Instructor
Computer Science (CSC142)

Course Description
Introduces computer programming, including data types, expressions, input/output, control structures, functional and object-oriented programming, and simple data structures. The course emphasizes problem solving skills through program refinement, documentation and programming style. **Prerequisite:** MAT102 (Intermediate Algebra) **Credit Hours:** 4

Resources
- **Books:**
  Savitch, Walt and Carrano, Frank M. *Java: An Introduction to Problem Solving & Programming*, 2009, Pearson/Addison-Wesley (required text)
- **Web sites:**
  Kumar, Amruth. Web-based tutors for C++/Java Programming, [http://acharya.ramapo.edu/~amruth/tutors/m06/kcc/home.html](http://acharya.ramapo.edu/~amruth/tutors/m06/kcc/home.html)
  Java 2 Platform Standard Edition 5.0 API Specification, [http://java.sun.com/j2se/1.5.0/docs/api](http://java.sun.com/j2se/1.5.0/docs/api)
- **Software:**
  Java 2 Platform Standard Edition (J2SE 5.0)
  Text editor and/or IDE (e.g. BlueJ)
- **Hardware:** personal computers

General Course Objectives
- Students will learn basic concepts of computer science and programming
- Students will write programs using Java
- Students will read, interpret, trace, extend, and debug program code
- Students will document programs thoroughly and succinctly
- Students will be able to explain and apply object-oriented programming techniques

Detailed Objectives
Unit I: Basics
At the conclusion of this unit, students will be able to:
- Name and describe the major hardware components of computers
- Describe how data and instructions are stored in a computer
- Define the following terms:
  - Program
  - Programming language
  - High-level language
  - Low-level language
  - Syntax
  - Semantics
- Explain the following concepts in an object-oriented programming context:
- Objects and classes
- Methods and messages
- Inheritance, superclasses and subclasses

- Differentiate between class and instance methods and data values
- Describe the software development cycle
- Identify the basic components of a Java program
- Describe the process of writing and running Java programs
- Define the following terms:
  - Compiler
  - Interpreter
  - Java Virtual Machine
  - Source code
- Write Java programs containing a single class and a single method
- Instantiate objects and call methods from standard Java packages
- Design and develop programs that solve simple problems using an incremental approach
- Write appropriate internal documentation
  - Declare meaningful identifiers
  - Write informative comments
  - Use consistent style with regard to program layout – indentation, use of spacing, etc.
- Explain the concept of data type
- Declare variables and manipulate values using Java’s primitive data types
- Write and evaluate arithmetic expressions in Java using literal values, primitive-type variables, and Math class methods
- Compare and contrast memory allocation mechanisms for objects and primitive types
- Use standard Java classes to perform programming tasks:
  - Input and output
  - Format numeric data
  - Data type conversion
  - Generate random numbers
  - Manipulate String data
  - Create simple applets that draw pictures

Unit 2: Control structures

At the conclusion of this unit, students will be able to:
- Read and write Java code containing selection control structures
- Implement selection control in Java programs using if, if/else, nested if/else and switch statements
- Read and write logical and relational expressions
- Recognize equivalent logical and relational expressions
- Evaluate Boolean expressions correctly
- Describe how objects are compared
- Choose appropriate selection control structures for given tasks
- Use selection control structures to solve programming problems involving comparison and decision-making
- Read and write Java code containing loops
Implement loop control in Java programs using while, do/while and for statements
- Read and write methods containing nested loop control structures
- Choose the appropriate loop control structure for a given task
- Recognize, explain, correct and prevent looping errors, including off-by-one errors and infinite loops
- Use loop structures to solve programming problems involving repetition
- Define the following terms:
  - Count control
  - Sentinel control
  - Priming read
  - Pretest loop
  - Posttest loop
- Read and write programs using combinations of selection and looping structures, including nested structures
- Trace execution of code containing looping and selection structures
- Test and debug programs using selection and iteration structures
  - Select appropriate test data
  - Fully exercise code

Unit 3: Classes and methods
- Define classes containing multiple methods
- Explain the difference between public and private methods and data members, and use each appropriately in programs
- Define and use void and value-returning methods
- Differentiate between local and instance variables, and use each appropriately in programs
- Trace execution of programs using multiple methods
- Pass primitive and object parameters to methods
- Write class constructors
- Define the following terms:
  - Overloading
  - Default constructor
  - Encapsulation
  - Mutator
  - Accessor
- Write methods that return objects
- Describe the uses of the keywords this and super
- Differentiate between reference parameters and value parameters
- Define class methods and variables
- Write simple recursive methods

Unit 3: Data structures and advanced topics
At the conclusion of this unit, students will be able to:
- Manipulate collections of data values using arrays of primitive data types or objects
- Define a method that accepts an array parameter
- Define a method that returns an array
- Write programs that store and manipulate data using two-dimensional arrays
- Describe two-dimensional array implementation in terms of an array of one-dimensional arrays
- Improve code reliability by incorporating exception-handling mechanisms
- Implement try-catch blocks
- Implement multiple catch blocks
- Define the following terms
  - Exception
  - Checked exception
  - Unchecked exception
- Write methods that throw exceptions
- Write programs that read and write files
- Use FileInputStream and FileOutputStream to perform low-level file i/o
- Use DataInputStream, DataOutputStream, ObjectInputStream and ObjectOutputStream to perform high-level file i/o on primitive data types and objects
- Read text files using a Scanner or a BufferedReader
- Write a text file using PrintWriter
- Create a simple GUI application using Java Swing objects
- Write event-driven programs

I. Instructional Methods

- Lecture/Discussion: Class meets four times per week for one hour at a time; the primary method of presentation during these sessions is lecture/discussion.
- Structured Labs: During regular class time, students are given problems to work out using computers and programming tools; structured labs occur approximately every week throughout the semester
- In-Class Exercises: Students are given opportunities throughout the semester to practice problem-solving skills through brief individual and group exercises designed for immediate practice of concepts demonstrated in lecture
- Homework: Students gain proficiency with the programming language and mastery of the concepts taught in class through a series of homework assignments in which they write original programs given problem specifications

Evaluation

- Exams and Quizzes 60%
- Homework Problems and Programs 40%