CSC 220 Data Structures

Prerequisites: *grades of C or better in MATH 226 and CSC 210. MATH 227 may be taken concurrently.*

Catalog Description: Linear and non-linear data structures, including lists, stacks, queues, trees, tables and graphs. Recursion, iteration over collections, sorting, searching, Big O notation and hash table.

Objective: This course uses Java programming language to illustrate linear and non-linear data structures, including lists, stacks, queues, trees, tables and graphs. Also covered are recursion, iteration over collections and sorting and searching topics, including Big O notation and hash table.

Course Topics:

- Object-oriented programming: Object-oriented design; encapsulation and information-hiding; separation of behavior and implementation; classes, subclasses, and inheritance; polymorphism; class hierarchies; collection classes and iteration protocols; generic types;

- Recursion: The concept of recursion; recursive specification of mathematical functions (such as factorial and Fibonacci); simple recursive procedures (Towers of Hanoi, permutations, fractal patterns); implementation of recursion

- Introduction to computational complexity: Asymptotic analysis of upper and average complexity bounds; big-O notation; standard complexity classes; empirical measurements of performance

- Fundamental computing algorithms: O(N log N) sorting algorithms (Quicksort, heapsort, mergesort); hashing, including collision-avoidance strategies;

- Abstraction and implementation of classic data structures: lists, stacks, queues, priority queues, hash tables, graphs, trees & balanced trees and dictionaries

Programming Exercises:

There will be a number of intensive hands-on exercises. Work must be done on an individual basis. Programming assignments will involve using the various data structures presented during the term.

Method of Evaluation:

Student learning will be evaluated on the basis of

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

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

Learning Outcomes:

At the end of this course students will

- Be able to write Java programs and an integrated development environment
- Utilize a debugger when doing software development
- Apply Data Structures and ADT concepts effectively when developing small to medium sized projects
- Write robust code utilizing exception handling language features
- Learn what and how to document each program project

Missed Exams:

Generally, there will be no make-up exams and no incomplete grades given. If you miss an exam, you must notify me before the exam or, if physically impossible, soon after. If any of the scheduled exam dates are in conflict with your religious observances, you must notify me, in writing, at least two weeks in advance of the exam. If you have an acceptable, documented excuse, you may be given a make-up exam or be given the average score of other exam(s) at the discretion of the instructor. Note that a make-up exam will consist of questions/exercises that might have a degree of difficulty that does not match those on the original exam.

Students with disabilities:

Students with disabilities who need reasonable accommodations are encouraged to contact the instructor. The Disability Programs and Resource Center (DPRC) is available to facilitate the reasonable accommodations process. The DPRC is located in the Student Service Building and can be reached by telephone (voice/TTY 415-338-2472) or by email (dprc@sfsu.edu).

Attendance:

“Students are expected to attend classes regularly because classroom work is one of the necessary and important means of learning and of attaining the educational objectives of the institution.” (SFSU Bulletin) To this end, attendance will be taken at different times through the term. Students missing class on a day of attendance will lose attendance points.

Required Text: *Data Structures and Abstractions with Java, 2nd Edition, by* Frank M. Carrano

Reference Text: A good Java reference

Created and approved: Novemer 2009
Revised: September 2012