CS 241 Data Structures and Algorithms I
Fall, 2007 Dr. Joseph Bergin

Text: Data Abstraction and Problem Solving with Java
Walls and Mirrors, 2nd edition
Frank M. Carrano, Janet J. Prichard
Addison Wesley, 2005
OPTIONAL:
Java Generics and Collections
Maurice Naftalin, Philip Wadler
O'Reilly, 2006

Dr. Bergin's Information:
jbergin@pace.edu
http://csis.pace.edu/~bergin
AOL IM: jb605pace

Dr. Bergin's Office Hours:
Office hours: Tuesdays 12:45- 1:15 and 3:30 pm – 5 pm, Thursdays 12:45- 1:15 and 3:30 pm - 5:45 pm, by appointment & online by AOL IM

Course Objectives and Outcomes
Objective 1 . Students will learn to design software using abstract data and control structures. These structures will include lists, stacks, queues, trees, and hash tables.

Outcomes : Students will demonstrate ability to:
a. Write and debug Java code for each of the above data types.
b. Use the above data structures in a programming application.
c. Extend the above data structures using inheritance to produce specialized structures.
d. Students will be able to give different design solutions to problems.

Objective 2. Students will learn to use recursion in program construction.

Outcomes: Students will demonstrate ability to:
a. Diagram backtracking on recursive tree.
b. Implement recursive algorithms for tree traversal.

Objective 3. Students will learn to implement abstract data types in alternate ways.

Outcomes: Students will demonstrate ability to:
a. Implement stacks and queues as both arrays and linked lists.
b. Implement stacks and queues using classes from the Java Collections class
c. Implement hash tables using different methods of probing.

Objective 4. Students will learn to quantitatively evaluate alternative implementations and explain the trade-offs involved.

Outcomes: Students will demonstrate ability to:
a. Evaluate the time complexity of fragments of code (e.g. sequential for loops, nested for loops)
b. Use Big-O notation to evaluate different implementations of stacks and queues

c. Use Big-O notation to evaluate different searching algorithms (linear, binary search on a sorted array, binary search tree, hashing –linear probing)

d. Discuss time/space trade-offs using Big-O notation.

Objective 5. Students will work on a group project.

Outcomes

a. Students will appreciate at least four phases of software engineering: specification, design, coding and testing.

b. Students will learn the benefits and the hindrances of working in a group.

c. Students will learn the difficulties of working on a larger programming project and learn the joy of getting a more complex program to run.

d. Students will realize the amount of testing that should be performed on a programming project.

Course Outline

I Principles of Programming and Software Engineering - Chapter 2
- Object Oriented Design
- Inheritance revisited - Chapter 9
- Linear and Binary Search
- Algorithm Efficiency, Big oh notation - Chapter 10

II Recursion: The Mirrors Chapter 3
- Recursion and efficiency

III Data Abstraction: The Walls Chapter 4
- Abstract data types including generic collections
- Java interfaces and exceptions
- Java Collections Framework (will be studied throughout the course)

IV Review of linked lists - Chapter 5
- Adding two polynomials
- Circular and doubly linked lists

V * The ADT Stack - Chapter 7
- Array and linked list implementation
- Solving problems using stacks

VI *The ADT Queue - Chapter 8
- Array and linked list implementation
- Simulation

VII *Recursion – Chapter 6
- Using and removing recursion
- Backtracking
- Divide and conquer algorithms
- Learning when to use recursion
- Studying the classic examples Eight queens, Knight's tour, permutations

VIII *Binary Trees –Chapter 11
Binary Search Tree and traversals
Binary Tree and traversals (Binary expressions and polish notation)
if time permits, balanced search trees

IX Test Driven Development

Evaluation (This is negotiable)
The primary determinant of your grade will be based on frequent assignments. There will also be a final exam on the required end-of-semester date. The exam is valued at approximately twice an individual assignment. Your grade is the simple average of all work required, including the exam.

Letter grades are assigned in the following way:

A, 90-100
B 80-90
C 70-80
D 60-70
F 0-60

Any assignment that receives a grade that the student doesn't like, may be redone. It will be regraded and the new grade substituted. This policy may need to be limited later in the semester and will end two weeks prior to the end.

Attendance will be taken at every class. If you must be absent, it is your responsibility to get the notes and homework assignment from another student before the next class.

It is likely that at least one of the programs assigned will be completed as a group project in order that you may study something of software engineering. There are advantages and disadvantages to a group project; you will learn them quickly.

Office hours 163 Williams St. 2nd floor. 212 346 1499 (DO NOT leave a phone message, use email instead). Contact me by email rather than by phone message.
Tuesday PM Thursday PM (as above)
I am not on campus other days.
You are welcome to visit me whenever my office light is on; that will probably be Thursday afternoons
email jbergin@pace.edu
AOL IM sessions are possible (jb605pace is my handle there when I'm available)

Course WIKI: http://csis.pace.edu:8097. Do not provide any web-visible link to this site, though you may bookmark it.

Your work must be turned in in hard copy at the beginning of the class at which it is due. By turning it in you certify that it is your own work except as noted by you on the work itself. Programming assignments (Java code) must be correctly formatted according to the course style guide.
See: http://csis.pace.edu/~bergin/patterns/codingpatterns.html