CSCI 2125 Section 001 -- Data Structures Spring 2008

Catalogue Description

The use and manipulation of structured data objects such as stacks, queues, trees, and lists; storage allocation for structured data objects.

Prerequisites

A grade of C or better in the following courses:
- CSCI 2120 Software Design and Development II and its lab CSCI-2121
- MTH 2721, Introduction to Discrete Structures.

If you do not meet these prerequisites, you **MUST drop,** unless you meet with me to show me (not just tell me) reasons why you are certain you have the knowledge those prerequisites fullfill.

Text:

An Introduction to Programming and Object Oriented Design using Java 1.5, 3rd Edition
J. Niño and F. A. Hosch, Wiley, 2008.
Data Structures and Problem Solving Using Java
Mark. A. Weiss. Addison Wesley, 3rd. Ed.

Other optional book:
Java Generics and Collections
Naftalin, M. and Wadler P. O'Reilly Media. Oct/2006.

Organization

The course is a continuation of CSCI 2120, and further develops the topic of data abstraction as the principal tool for system design. Several important subjects will be discussed, including:
- **Design**: we will examine design strategies for library **API**, and try to categorize some of design approaches
- **Data structures**: we will examine the specification and alternative implementation of a number of important classes designed to contain collections of objects, so called *container* classes. We study **linear** containers *List*, *Stack* and *Queue*; examine **non-linear** data structures such as trees. We will also study the Hash table abstraction and its implementation using linear as well as non-linear containers. For each of the containers study we will explore different implementation and analyze their trade-offs. We will also consider questions of storage allocation for containers.
- **Algorithms**: Study several algorithms for containers, among those sorting and searching as well as many others.
- **Efficiency**: we will spend some time becoming familiar with the notion of computational complexity, and informally determine the complexity of methods of the containers of study.

Objectives
- Design, implementation, and use of Container classes.
- Use of generics for the design and implementation of Container classes and their algorithms.
- Complexity analysis of algorithms and container memory use.
- Introduction of design patterns in the design of containers and algorithms.
- Design and implementation of algorithms for containers.
- Understand the Java Collection API.
- Analysis and design of a library API: principles, design and implementation choices.

Outcomes
- Understand the following concepts:
  1. Collection of data values and the classes to model them, called Container classes.
  2. Class inheritance and its relation with Container classes.
  3. Container equality
  4. Independence of the Container from the type of the objects it contains.
- Know to design and implement:
  1. Generic container classes.

2. Generic algorithms on containers.

- Design, implementation and use of Linear and Non-linear containers.
- Be able to analyze the complexity of algorithms for containers.
- Be able to measure the memory footprint of a container.
- Be able to use patterns in the design and implementation of containers and its algorithms.
- Be able to design/implement sorting algorithms using containers.
- Be proficient with the use of Java Collection API classes.

Office Hours

Office: MATH 331.
Office hours:3:30-4:30 pm. T, Th. Please try to come by then, or make an appointment.
Phone: 280-7362

Email/Home Page

Email: `jaime@cs.uno.edu`
The course page: log in to: `http://uno.blackbord.com` using your UNO LAN account.
From time to time I will communicate via email using your UNO lan email account or your department account; you are responsible to read your email in a timely manner. Also, if you send me email

- do it using your department account or your UNO account.
- write as subject: CSCI-2125 student.

Do not expect me to open nor answer any student email that was not sent through your department or UNO account.

Exams

We will have 2 tests and a final. The tests are **tentatively** scheduled for October 2, November 18th. The final exam is scheduled for Tuesday dec 9, 5:30-7:30pm. in the same classroom. No makeup exams will be given.

Homework

A number of programming homework (about 6) will be assigned during the semester. Since I may sometimes test programs by running them under your departmental account, you should be prepared to make executable versions of your programs locally accessible. You are also required to turn in hard-copy pretty-printed source.
Assignments are due at class time, on the expected date, and should be handed in to me, or at your risk, left in my mailbox in MATH 311. The box outside my office is an **OUT** box, please **do not** place any work there. **No late** submissions accepted. If for some reason I take your work late. I generally grade late homework **MUCH** more rigorously and will be grading it based on 75% of grade, and grade it at my convenience; that means that you will not get it back until after the end of the semester. I STRONGLY RECOMMEND THAT YOU SUBMIT BY HOMEWORK DUE DATE WHATEVER YOU HAVE DONE TILL THAT POINT.
No late homework will be accepted during or after the last week of classes. *If you fail to submit a homework, you will fail the course.*
Unless you are **explicitly** instructed otherwise, homework assignments are individual projects. Collaboration is considered cheating, and incorporation of un-referenced ideas or materials that are not your own is considered plagiarism. *You should clear with me use of any code not developed by you, and such code should have a reference for its source*. You will not receive credit for code not developed by you.
Note: it is better to turn in an assignment and get a poor grade than not to turn it in at all and fail the course. Also note that generally a program must work correctly to be "minimally satisfactory". Turning a program *that works* insures you of at least a C, not of an A.

You will receive an automatic F in the course if you fail to submit a programming homework.

OUT box

I will return homework graded as soon as possible. Absent students will find homework in out box.

Grading:

- Homework, pop tests: 40%
- In class Tests: 30%
- Final: 30%

Letter grades will be assigned as [90 - 100] = A, [80 - 90) = B, etc.
If Final exam grade >= 80% (B) it will replace the lowest in class test grade, if test grade lower than final grade.

Auditors

If you are auditing the course, you must attend class regularly and take the examinations to receive a satisfactory audit.

Attendance

Attendance is mandatory. Failure to attend will affect your final grade, including failing the course. Every absence must be justified in writing. You will automatically fail the course if you accummulate 3 consecutive unjustified absences, or 6 absences (justified or unjustified) in total. Unjustified absences will take up to 10% from your final grade.

Cheating

Finally, I must call your attention to the University's policies regarding academic dishonesty. (See pages 44-47 of the Student handbook.) Academic dishonesty includes cheating, plagiarism, and collusion. In particular, it includes "the unauthorized collaboration with another person in preparing an academic exercise" and "submitting as one's own any academic exercise prepared totally or in part for/by another." In the event of academic dishonesty, in its first occurrence **the students involved will be assigned a grade of 0** on the work involved, and students will be informed in writing of the action taken, and a copy of this letter will be sent to the Assistant Dean for Special Student Services. For any other occurrence, students will automatically fail the course and will be reported to the Student's judicial office.

Last day to drop : November 10, 2008