Illinois State University

ITK 179, Fall 2007

Introduction to Data Structures

MW STV-108 2:00~3:15 PM

Instructor: Chung-Chih Li, Ph.D.

Office: Old Union 105

Office Hours: MTWT, 11:00 AM ~ 12:00 PM or by appointment

Contact: Tel:(309) 438-7952, Email: cli2@ilstu.edu

★ ★ ★ ★ ★ ★ ★

WebPage of the course: `http://www.itk.ilstu.edu/faculty/chungli/ITK179`

Students should regularly check the webpage for important information about assignments, data, due dates, lecture notes, and announcements. *However, announcements made in the class should be considered as official if they are not consistent with the ones on the webpage, since I may not be able or remember to update every announcement.*

Prerequisites:

ITK 168 with a C or better grade. This is a continuous course of ITK 168. We assume that students already have basic skill in developing easy JAVA programs including using some IDE, primitive data types, control structures, and basic I/O of the language.

Note: *Students will not receive credit if they have taken ITK 169 or ITK 275.*

Textbooks:

Data Structures in JAVA: From Abstract Data Types to the Java Collections Framework, by Simon Cray, Pearson/Addison Wesley, 2007.

Reference:

Objects, Abstraction, Data Structures and Design Using JAVA, Version 5.0, by Elliot B. Koffman & Paul A.T. Wolfgang, Wiley 2005.

Java, Software Solution – foundations of program design, 4^{th} edition, by John Lewis and William Loftus, Addison Wesley, 2004

Course Description and Purposes:

In this course, we will further strengthen your programming skill and learn more problem-solving techniques through studying some advanced features and concepts of the programming language – Java. We will emphasize on both traditional data structures and the relatively new programming paradigm – object-oriented design.

The data structures we are going to learn include *linked list*, *stacks*, *queues*, *trees*, and *binary search trees*. We will organize our data into those structures and see how those data structures improve our programs' performance. We will also learn how to measure the computational complexity of a program or a particular problem. Sorting is a very important task in computer science and we will scratch this problem and some easy solutions in the class. Also, we will learn a powerful programming technique – *recursion* – and understand it as a natural approach to solving complicated problems.

We assume you already have some basic concepts of Java programming from ITK 168. Nevertheless, at the beginning of the semester, we will briefly review the basic yet necessary features of Java and explore you to more Java IDE including Eclipse and NetBeans. We will scratch the surface of UNIX and run Java programs from the *command line* mode as the default programming environment throughout the semester. Issuing commands under *command line* to

operate an OS is primitive but yet the most direct way to use the OS, since working under the command line mode can minimize the hidden tasks run behind the scenes. Thus, we encourage you to develop and test your Java programs under the UNIX command line mode and be familiar with the basic UNIX's commands for your feature study. However, you can use any IDE to develop your programming assignment.

Objectives of the Course: After this course, students should:

1. Understand and be able to implement the basic data structures: *linked list*, *stacks*, *queues*, *trees*, and *binary search trees*.
2. Understand recursion and be able to solve some typical problems recursively.
3. Be able to design, implement, test, and debug moderately complex Java programs.
4. Be familiar with the paradigm of Object Oriented Programming: *Classes, objects, methods, class inheritance* and *hierarchies,* and *polymorphism.*
5. Be ready to study more advanced algorithm analysis.
6. Be comfortable to explore to any new IDE in the feature.
7. Be comfortable to explore to Unix in the feature.

Examinations: (350 points)

Two midterms and one final exam; 100 points for each midterm and 150 for the final

- Unless announced otherwise, all tests are accumulative, closed book, and indispensable. No makeup test will be given unless a documented absence is authorized by the university.
- Every student is allowed to bring a *self-prepared hand-writing* crib sheet to the test. You can **write** down anything on both sides of **one** letter-sized paper. No circulation during the test.

6^{th} week's	Midterm I	100 points	September 24, Monday
11^{th} week's	Midterm II	100 points	October 29, Monday
17^{th} week's	Final Exam	150 points	Dec. 13, 1:00 – 3:00 PM, Thursday.

Pop quizzes:

Some pop quizzes may be given without notice in advance. Each quiz carries 10 points towards students' final scores. There are considered as the bonus points. The coverage of every quiz is also accumulative, including the materials that are three-month-old and those covered in the class right before the quiz. A typical quiz takes about 10 minutes. No makeup quiz will be given if missed. If you miss a quiz due to a university authorized absence, we will use the average of your rest quizzes as the score; otherwise, you get a 0 for the missing quiz.

Programming Assignments: (500 points)

About 8 programming assignments will be given throughout the semester; some will be teamwork. Teams will be form randomly upon each assignment. Each team should submit one report and every one in the same team will receive the same score. The weight of each assignment depends on the degree of its difficulty. One should follow the following guideline for submission:

Strictly follow the guideline for submission

- Put a few comment lines at the beginning of your program files, in which you should clearly indicate every team member's name and ID and claim the copyright. Also, at the beginning of each method put comments with the name who actually implemented the method. **Student who fails to do so will receive 0 point on the assignment.**
- Submit your assignment with items as described in the following order.
  1. A cover page with names and student ID's in the team.
  2. A brief summary about the assignment and your approach to the problem. You may include the difficulties you had faced, if any, or why you think your program doesn't work. It is very common and not a shame to admit that your program doesn't work under time constraints, but a reasonable self-diagnosis deserves reasonable partial credit.
  3. A hard-copy of the source codes.
  4. A hard-copy of the directly output of your program, if any.
  5. All items described above must be put in a **letter-sized Manila folder** with your name on it.
  6. Use the web submit described as follows to submit your programs.

Web Submit

All programs and programs only should be submitted through Web Submit before the class of the day date.

1. Link to `http://kirk.itk.ilstu.edu`
2. Click on `Submit Your File`
3. Enter `stu179` as ID and `*******` as password. Select `ITK179` and `01` for course number and section number, respectively. Then click `Login`
4. Enter `XXXXX` as the ULID, where `XXXXX` is your or your team partner's ULID. Enter x as the project number, where x is the number of the assignment. Use `Browse..` to select the file you want to submit. Then click `Submit`. If you have more than one file to submit, repeat this step as necessary.

Team Work: If the assignment is required to work by teams, this is how you split the work. Each team for a programming assignment consists two students. Each person has to implement some codes equally divided. One should write up the summary of the approaches and algorithms used and the other should explain the output of the program, i.e., why we obtain the results according to the implementation. In your report, you have to indicate the person who did the job. However, **both of you will receive the same score**.

Each team has to submit only one report and one set of programs.

Try very hard to avoid the following troubles:

1. Any form and any degree of plagiarism will receive 0 point.
2. If your program contains syntax error, you will receive 0 point.
3. If the hard-copy of the direct output of your program is inconsistent to your program's design, you will receive 0 point. This is a kind of cheating.

Attendance: Attendances will be taken impulsively. Each unauthorized absence will cost you 20 points from your score tally.

Class Participation : (150 points)

This includes pop quizzes, attendance, class behaviors, helping peer students inside and outside the classroom.

Academic Honesty:

Cheating, plagiarism, collusion, abuse of resource materials, and their consequences are defined and described in *ISU 2006-2007 Undergraduate Catalog*, Section: Academic Policies and Practices, Article: Academic Integrity (Page 63) and *Code of Student Conducts* under `X.C.` Disciplinary Bodies And Procedures – Academic Honesty Cases. Students giving away academic works for assignment offered for credit to other students working on the same assignment will be considered as guilty as academic dishonesty, and will receive the same penalty. More information can be found at:

`http://www.deanofstudents.ilstu.edu/crr/downloads/Code_of_Student_Conduct.pdf`

Grading Policy:

The perfect score is 1000. Your grade is based on the percentage of the total points you receive according to the following scheme.

Percentage of total points	Grade	
85 %	A	Excellent
75 %	B	Good
65 %	C	Satisfactory
50 %	D	Passing
- - -	F	Failure

I do not curve!!

I'm not afraid to give all A's, neither am I to give all F's. Thus, you don't have to knock down your friends to get a good grade. In other words, you can't hide behind someone else, because you two could be both shot down. So, do help your classmates if they need you.

Play Ball!!

Tentative Topics and Schedule:

Keep the following table of tentative topics and schedule handy, and try to keep up with the schedule. Read the assigned materials before the class.

Week	Topics	Reading
1: Aug. 20	Introduction to Java and Unix	Chapter 0, 1
2: Aug. 27	OOP, ADT, a case study.	Chapter 1
3: Sep. 3	(Sep. 3, Labor Day, No class) Algorithm Analysis	Chapter 2
4: Sep. 10	Fundamental Data Structures, Array and Linked Lists	Chapter 3
5: Sep. 17	Collection Classes	Chapter 4
6: Sep. 24	**(Midterm 1)**, List	Chapter 5
7: Oct. 1	List	Chapter 5
8: Oct. 8	Stack and its applications	Chapter 6
9: Oct. 15	Stack and its applications	Chapter 6
10: Oct. 22	Queue	Chapter 7
11: Oct. 29	**(Midterm 2),** Recursion	8.1 $\sim$ 8.3
12: Nov. 5	Recursion	Chapter 8
13: Nov. 12	Sorting and Searching	Chapter 9
14: Nov. 19	(Thanksgiving week, no class)	
15: Nov. 26	Tree and Binary Search Tree	Chapter 10
16: Dec. 3	Binary Search Tree, Catch-up week	Chapter 11
17: Dec. 10	**Dec. 13, 1:00 – 3:00 PM, Thursday, Final Examination.**	