

Instructor Dr. Lixin Tao, ltao@pace.edu, <http://csis.pace.edu/~lixin>
Pleasantville Office: G320, (914)773-3449

Lectures PLV Goldstein 315, Tuesdays and Thursdays, 6:00PM-7:50PM

Office Hours Tuesdays and Thursdays, 4:00PM-5:30PM and 7:50PM–9:30PM at PLV G320

Prerequisite Ability to specify classes in Java, including constructor overloading, private fields, public instance methods, and class variables and methods; ability to manage one-dimensional arrays of types **int**, **double**, and **String**

Description

Coverage of linked lists; recursion; derivation including constructor chaining, abstract classes, and polymorphism; interfaces; exception handling; data structure implementation for stack, queue and priority queues; event-driven and GUI programming; multithreading and thread synchronization; socket-level networking; and basics of database programming.

Textbook

- *Introduction to Java Programming: Comprehensive Version, 5th Edition*, by Daniel Liang. Prentice Hall 2005. ISBN: 0-13-148952-6. (book resources at <http://www.cs.armstrong.edu/liang/intro5e/intro5estudentsolution.html>)
- *Sun Java Tutorial* (<http://java.sun.com/docs/books/tutorial/>)
- *Class notes*

Learning Objectives and Expected Outcomes

Objective 1: Acquire the ability to build and manage linked lists of objects – objects with a “next” field such as:

```
class Node
{
    char letter;
    Node next;
}
```

This reinforces the understandings: (i) that a logically unlimited number of objects, each with an independent set of instance variables, may be instantiated from the same class; (ii) that a reference variable stores an address; and (iii) that diagrams provide guidance in building and desk-checking code.

Outcome: Students will be able to construct classes for list processing: handling a singly linked list as a stack, as a queue, and as a priority queue.

Objective 2: Learn about recursion (methods containing one recursive call), the concept of a stack of local referencing environments, work “on the way up” (i.e. the iteration of code before the recursive activation), work “on the way down” (i.e. the iteration of code that follows the recursive activation), and tail recursion.

Outcome: Students will be able to desk-check recursive methods by sketching the stack of activations and returns, and students will be able to build recursive methods such for processing a singly linked list from back to front.

Objective 3: Learn the principles associated with the construction of class derivation hierarchies. This includes the inheritance of public members from ancestor classes, augmenting an extending class with incremental fields and methods, method overriding, the difference between overriding and overloading, the principles of constructor chaining, polymorphism, and abstract classes. This also includes the uses of **super**, **final** methods and **final** classes, as well as overriding **equals()** and **toString()**.

Outcome: Students will be able to explain and illustrate how polymorphism enables “old code” to manage new objects as well as to describe and exemplify the other programming constructs used in assembling classes into object-oriented software systems.

Objective 4: Learn about the **interface** construct, the use of an interface type for implementing callbacks, and the use of the **java.util.Iterator** interface (this includes the construction of an inner class and the concept of a design pattern).

Outcome: Students will be able to specify an **interface**, declare classes that implement that interface, write code in accordance with the rules for interface usage, and describe and exemplify the value of this construct.

Objective 5: Acquire the ability for exception handling. This includes the *try/catch/finally* statement, the **throws** clause, extending the **Exception** class for checked exceptions and the **RuntimeException** class for unchecked exceptions, the **throw** clause, and the propagation of exceptions.

Outcome: Students will be able to write programs that recover from exceptions thrown by Java as well as throw and recover from exceptions of their own declaration.

Objective 6: Acquire the ability to use arrays of objects of any type, and, from the Java Collections Framework (in the **java.util** package) to use objects of type **Vector**, **ArrayList**, and **LinkedList**. This includes object up-casting and down-casting, wrapping primitives for storage as objects (as well as unwrapping), slicing, and the adapter pattern.

Outcome: Students will be able to implement customized classes for stacks, queues, and priority queues using the adapter pattern and the linear data containers supplied by Java.

Objective 7: Understand the concept of event-driven programming and master the basic skills in GUI programming with SWING.

Outcome: Students can design and implement applications with natural graphical user interfaces.

Objective 8: Understand the concepts of processes and threads and master the basic skills in spawning multiple threads and synchronizing them in multi-thread applications.

Outcome: Students can design and implement responsive and efficient applications with Java multi-thread programming.

Objective 9: Understand the concept of client-server computing and master the basic skills in Java socket-level programming.

Outcome: Students can design and implement client-server applications that can interactive dynamically over the Internet.

Quizzes

There will be two one-hour open-book quizzes on both fundamental concepts and writing complete programs. The quizzes will be on February 23 and April 18 respectively. No makeup quizzes except for students with instructor approval beforehand.

Programming Assignments

There will be three programming assignments due on February 21, March 14, and April 11. The source code of answers to the assignments must be submitted through Blackboard by the due date. Each day of late submission will incur a 10% penalty on the grade for the assignment, and under no circumstances, except for cases with instructor approval beforehand, an assignment submission will be accepted after five days past the deadline.

Course Project

From week 2, students will set up teams of 4-5 people each. Each team will elect a team leader responsible for team activity coordination. Over the semester, both teams will design and implement a client-server application named *Pace Instance Messenger (PIM)*.

Team 1 will be responsible for the server-side design and implementation, which includes exposing the PIM service on the Internet for accepting client participation requests, spawning new threads to serve different clients, processing client events and supporting the business logic of PIM, logging messages in a database, and answering client database queries about the logged messages.

Team 2 will be responsible for the client-side design and implementation, which includes a quality graphic user interface for PIM, communicating with other clients through the PIM server, handling any networking exceptions, maintaining dynamically the list of current participants, and querying and displaying messages logged on the server that contain special key words.

On March 14, each team will take 30 minutes to present its application design. On April 27, each team will take 30 minutes to present its application implementation, and the two teams will provide a live demo of the complete PIM project together. The two teams will work together to complete a joint project report that should include sections for problem description,

design documents, implementation highlights, experimental study, known problems, installation guide, and user guide. The project report should be due on April 27 too.

Class Attendance

If a student is absent from a class without a justifiable and provable reason, he/she will suffer a 5% deduction in the final weighted total of his/her grades. A student missing four classes without justifications will fail the course.

Grading Scheme

Assignments	30% (10% each)
Quizzes	20% (10% each)
Final Exam	10%
Project oral presentations	20% (10% for design, 10% for demo)
Project report	20% (10 for technical quality, 10% for writing)

Important Dates

February 21: assignment 1 due
February 23: quiz 1
March 14: assignment 2 due; project design presentation
April 11: assignment 3 due
April 18: quiz 2
April 27: project final demo and presentation

CSIS School Policy Regarding Academic Integrity

1. Definition.

Students must accept the responsibility to be honest and to respect ethical standards in meeting their academic requirements. Integrity in the academic life requires that students demonstrate intellectual and academic achievement independent of all assistance except that authorized by the instructor. The following constitute academic dishonesty. The list is not inclusive.

- a) Exams
 - i) Copying from another student's exam.
 - ii) Deliberately allowing other students to see and copy from your exam.
 - iii) Using notes or calculators without permission from the professor or proctor.
 - iv) Passing notes or calculators to other students without permission.

- b) Papers and projects
 - i) Copying others' writing without proper reference.
 - ii) Copying code or work from other students outside a team environment. This could be either from printouts and notes or from electronic media. This includes copying the structure of a program while changing cosmetic details such as identifiers and comments.
 - iii) Deliberately allowing other students to copy your code or work, again either from printouts, notes or from electronic media. (This does not preclude a student "helping" another on a project as long as it is limited to giving information/hints and not code/solutions.)
 - iv) Submitting a paper, program, or project that was done by someone else.
 - v) Collaboration with one or more other students without the prior permission of the instructor.

2. Consequences. The following consequences will be affected:

- a) The first student offense may result, at the discretion of the instructor, in penalties including a zero on the offending course work or an F for the offending course.
- b) The second student offense in any course may result in an F for the offending course.
- c) The third student offense in any course may result in dismissal from the University.
- c) The Dean's office shall keep a student record of all student offenses occurring in courses offered by the School of CSIS including the first offense. This record should be destroyed when the student graduates from the University. The record shall be associated with the student and not with any particular course.

3. Procedures for determining an offense. The following procedure will be used:

- a) If the student admits to the offense, the appropriate penalty shall be enforced.
- b) If the student contests the charge, the Chair of the department in which the course was offered will make a decision as to the facts of the case. If the professor is also the Chair, this step could be skipped.
- c) If the student disagrees with the Chair's decision, he or she may request a hearing from the *Undergraduate* or *Graduate Scholastic Standing Committee*, depending upon the student's status. The Committee shall make a recommendation to the Dean concerning the facts of the case.
- d) Both the professor and the student may submit to the Committee relevant information in writing. The professor and/or the student may also appear before the committee, but usually not concurrently. No others may attend the Committee hearing, but the Committee may also consider the written statement of witnesses and other concerned persons.
- e) The decision of the Dean shall be final.
- f) A confirmed student offense shall be entered into the student's record in the Dean's office.