Introduction to Computer Science

Instructor: Shimon Schocken

Teaching Assistant: Boaz Kantor

“What I hear, I forget; What I see, I remember; What I do, I understand.”

(Confucius, 551-479 BC)

Objectives: The course seeks to (i) Overview key ideas and techniques in computer science, and (ii) Teach object-oriented programming using Java. These two activities will go hand in hand: the theory and practice of computer science will unfold in the context of many programming examples and exercises. Through this experience, students will develop an appreciation of the elegance and beauty of computer science, and will become competent Java programmers. The course assumes no previous experience in math or programming.

Methodology: The course is based on 13 weeks of instruction. Each week consists of two 2-hour lectures, one 2-hours recitation (Targil), and a homework assignment. Of these three elements, the homework assignment is the most important activity. Some of these assignments will be quite challenging, and you should expect to spend many hours working on them.

The course grade is based on a midterm exam (15%), final exam (60%), and homework assignments (25%). Homework assignments must be submitted each week on or before the deadline. Late homework will not be graded and will automatically get a zero grade.

Textbook: "*Java Software Solutions: Foundations of Program Design*", by Lewis and Loftus, Published by Addison Wesley, Pearson International Edition, 5th Edition. (Previous and other editions of this book are OK also). The book is available at the IDC book, IDC library, and numerous web book stores. There are many other Java textbooks out there, and you are welcome to use any of them instead of this one.

The Java Tutorials: Provides a great way to learn Java on your own and augmnet what we do in this course. To find it, Google "Java Tutorials" and start with "Getting Started".

Book recommendation for advanced students: If you are already an experience Java programmer, the following book is highly recommended: *Effective Java*, by Joshua Bloch, Addison-Wesley, Second Ediction. To clarify: the material covered in this book is NOT part of this course and is intended only for people with a significant programming experience.

The course web site (www.intro2cs.com) is the hub of all the course activities and the only official channel of communications between the course staff and students. All lectures, recitations, homework assignments and course materials will be uploaded to the course web site. Nothing will be distributed on paper. It is the student's responsibility to download the relevant course materials from the web site and decide what should be printed and what can stay on-line. The course web site is also the repository of all the course's announcements, rules and regulations. Each student is responsible for visiting the course web site at least once a day and following the instructions listed there.

Questions and answers: are welcomed and encouraged. If you have any question about any aspect of the course, please post it on the "Q&A" section of the course web site. Your question will be answered within 24 hours.

Civilized behavior in and out the classroom is important and expected. Please refrain from doing things that would irritate you if you were to teach this course yourself. Examples of offensive behavior include being late to class, eating / reading / browsing in class, posting inappropriate or silly messages in the course web site, and so on. Use your judgment.

Course Plan (by week)

This plan listed below is tentative. The actual week-by-week plan will be published in the course web site, one week in advance. All reading references are to "*Java Software Solutions: Foundations of Program Design*", by Lewis and Loftus.

Date	Topic	Reading
Oct 11	**Wlecome to Computer Science:** The elegance, the beauty, the power.	
Oct 14	**Welcome to Java:** Introduction to object-oriented programming using Java.	Pages 53-75
Oct 15	Recitation	
Oct 18	**Basic program elements**: Variables, assignment, primitive data types, String types, expressions, user interaction,	
Oct 21	**Using Classes and Objects.** Working with abstractions and APIs, creating and using objects; Static methods, Classes of interest: `String`, `Random`, `Math`; Turtle graphics.	Chapter 2 Chapter 3
Oct 22	Recitation	
Oct 25	**Control Structures**: Boolean expressions, procedural programming, control structures: `if`, `switch`, `while`, `do`, `for`.	Chapter 5
Oct 28	**Control Structures**: Continued	
Oct 29	Recitation	
Nov 1	**Arrays I**: Declaring and using arrays; arrays of objects; Two-dimensional arrays; the `ArrayList` and `Vector` classes, Command-line arguments.	Chapter 7
Nov 4	**Writing classes I:** Classes and objects revisited; Class anatomy, encapsulation, method anatomy, parameter passing, constructors, overloading, visibility modifiers, static and auxiliary methods; Design issues.	
Nov 5	Recitation	
Nov 8	**Arrays II**: Declaring and using arrays; arrays of objects; Two-dimensional arrays; the `ArrayList` and `Vector` classes, Command-line arguments.	Chapter 4
Nov 11	**Writing classes II:** Classes and objects revisited; Class anatomy, encapsulation, method anatomy, parameter passing, constructors, overloading, visibility modifiers, static and auxiliary methods; Design issues.	
Nov 12	Recitation	
Nov 15	**Hardware fundamentals:** Computer organization, CPU, memory, bus, machine language, execution cycle, low-level programming, assembler.	Pages 29-53
Nov 18	**Hardware fundamentals**: continued	
Nov 19	Recitation	Chapter 10
Nov 22	**Software fundamentals**: High level programming, pseudo code, compiler, interpreter, virtual machines, byte code, operating system essentials, bootstrapping, browsers, HTML, applets.	

Nov 25	**Internal Data Representation** of numbers and text. Binary, octal, hexadecimal, and decimal representations; The 2's complement system; The floating-point system; ASCII and Unicode. Digital representations of graphics and other media types**.**	
Nov 26	Recitation	
Nov 29	**Midterm examination**	
Dec 2	**Algorithms I:** Induction, correctness proofs; Searching, sorting, divide and conquer, breadth first search & depth first search; Space and time efficiency; Intractable problems, complexity analysis.	
Dec 3	Recitation	
Dec 6	**Algorithms II:** Continued.	
Dec 9	**Algorithms III:** Continued.	
Dec 10	Recitation	
Dec 13	**Recursion**: Reduction definition, reduction techniques, recursive thinking, recursive algorithms, recursion in Java, runtime stack implementation**.**	Chapter 11
Dec 16	**Collections:** Abstract Data Types; Recursive data structures; Sets, stacks, linked lists, Iterators. Linear and dynamic implementations.	Chapter 12
Dec 17	Recitation	
Dec 20	**Collections:** Continued	
Dec 23	**Object-Oriented Design**: Identifying classes and objects, class relationships, abstract classes, interfaces, method design, enumerated types.	Chapter 6
Dec 24	Recitation	
Dec 27	**Inheritance**: Basic concepts, IS-A relationships, when to derive sub-classes, visibility issues, default constructors, `super`, code reusability, overriding rules.	Chapter 8
Dec 30	**Inheritance**: The `Object` class, class hierarchies. Widening and narrowing: sub-classes as sub-types, compile-time VS run-time types, virtual methods, `instanceOf`.	
Dec 31	Recitation	
Jan 3	**Polymorphism**: Late binding, polymorphism via inheritance, polymorphism via interfaces, designing for polymorphism, multiple interfaces; Sorting and searching examples**.**	Chapter 9
Jan 6	**Advanced Java programming:** Time permitting, we will explore one of these topics: graphics, events-driven programming, communications, multi-threading	