Lecture 0:
Introduction to Computer Science (and Java)

Course Website and Syllabus

The course website is located at http://cs.nyu.edu/~joannakl/cs101.06_s16/. It has the basic course information and the course syllabus. I will be posting all the lecture summaries and assignments there.

There is also NYU Classes (access it through NYU Home http://home.nyu.edu) page for the course. You will need to submit all the assignments through NYU Classes. I will post your grades there.

Software For the Course

Java Development Kit (JDK) http://www.oracle.com/technetwork/java/javase/downloads/index.html

In order to compile and run Java code you need the tools that come with the Java Development Kit. It is likely that you already have it installed on your computer, but if that is not the case go to the Oracle website and download the package appropriate for your operating system. You should be using Java 8 or 7 (see the link above).

Eclipse http://www.eclipse.org/downloads/

One can write, compile and run Java programs using the JDK, a simple text editor and a command line environment (terminal). Most beginners and many advanced programmers prefer to work with an intergrated development environment (IDE). Eclipse is one of several IDEs. We will use Eclipse in class demonstrations and the tutors can help you with it. See section 1.12 in the textbook for help on installing Eclipse and writing your first Java project.

Intro to Computers and Computer Science

What is a computer?

A *computer* is an electronic device that stores and processes data.

Parts of a computer:

- a central processing unit (CPU)
- memory
- storage devices
- input devices
- output devices
- communication devices

What is computer science?

From Wikipedia: ”Computer science is the scientific and practical approach to computation and its applications.”

From Merriam Webster dictionary: ”Computer science is the study of computers and their uses.”

What do computer scientists study/do?

- scientific computation, modeling, simulations
- artificial intelligence
- graphics design
- robotics
- game design / programming
- security, cryptography
- bioinformatics
- image processing (biomedical image processing)
- ...

Operating System

The *operating system* manages and controls a computer's activities. It acts as an intermediary between users and user programs on one end and the hardware on the other end.

Programming Languages

Machine language is the language ”spoken by” the hardware: usually sequences of zeros and ones (known as binary).

Assembly language is a low level language that humans can understand. It has English language based instructions that correspond to the operations that can be performed by the hardware.

High-level programming languages are hardware independent and are much easier for humans to understand. Examples: Java, Phython, ...

Java

Java language specification - technical definition of the language that includes the syntax and semantics of the Java programming language;
http://docs.oracle.com/javase/specs/

Application program interface (API) - predefined classes and interfaces for developing Java programs;
http://docs.oracle.com/javase/8/docs/api/

Java is a full-fledged and powerful language that can be used in many ways. It comes in three editions:

- Java Standard Edition (Java SE) can be used to develop client-side standalone applications or applets.
- Java Enterprise Edition (Java EE) can be used to develop server-side applications, such as Java servlets and JavaServer Pages. Java Micro Edition (Java ME) can be used to develop applications for mobile devices, such as cell phones.
- Java Development Toolkit (JDK) - a set of separate programs, each invoked from a command line, for developing and testing Java programs.

Integrated development environment (IDE) - software that provides/combines many tools for rapidly developing (Java) programs: editing, compiling, building, debugging, and online help are integrated in one graphical user interface. Examples:

- NetBeans,
- Eclipse,
- Dr. Java,
- TextPad.

A Simple Java Program

Listing 1: What does this code do: what does each line mean? what is the output?

```
public class Welcome {

  public static void main(String[] args) {

    System.out.println("Programming is fun!");

    System.out.println( (10.5 + 3 * 2.5) / (7.5 - 4.5/3 ) );
  }
}
```

Listing 1, version 2 :

```
// name of the class,
// every Java program needs at least one class
public class Welcome {
    // main method of the class Welcome,
    // that’s where the program starts
    public static void main(String[] args) {
        // display the string
        System.out.println("Programming is fun!");
        // compute and display the number
        System.out.println( (10.5 + 3 * 2.5) / (7.5 - 4.5/3 ) );
    }
}
```

Output:

```
Programming is fun!

3.0
```

Creating, Compiling and Executing a Java Program

action	**result**	**comment**
Create/Modify source code: use plain text editor	Source code files, ex. `Welcome.java`	high level language, human readable
Compile source code: `javac Welcome.java`	Bytecode, ex. `Welcome.class`	if errors occur go back to step 1 (compilation errors) low level language, similar to machine code, runs on JVM, plus: machine independent, minus: slower
Run bytecode: `java Welcome`		if errors occur go back to step 1 (run-time errors)