**UNIVERSITY OF MICHIGAN**
**EECS 282**

# Assignment 1 – Installing Java Software and Running a Program

Due Date: Tuesday, January 12th at 11:55 PM. The deadline is precise and the time is decided by ctools. You would be using late days for submission if ctools records the receipt as after the above time. We recommend submission a few hours before the above deadline to allow for slow response time from ctools. You can submit multiple times. Ctools only saves the latest submission

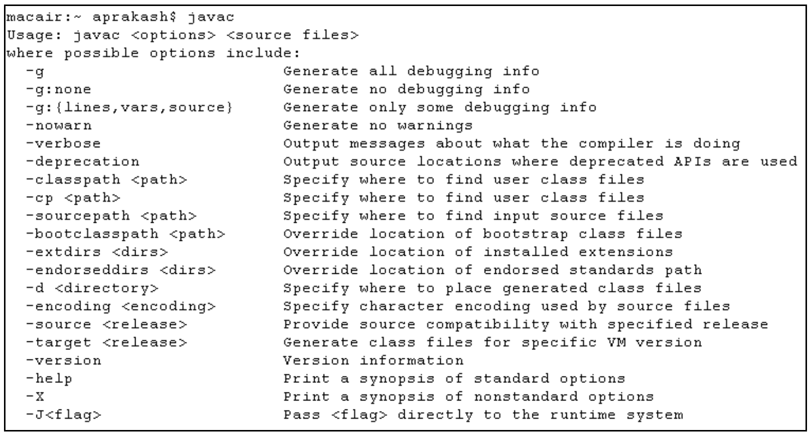
## Overview

This assignment consists of three parts. The goal is to get you started, motivate the problems we will be solving in the class, and to make sure that you have the proper software tools installed for the remainder of the course. Glance at "What to Submit" at the end when you start working on a part so that you know what information to save from each part.

## Part 1: Installing Java Software and run Java from Command-line

You should install Java software on your laptop or personal computer. Here is what you need. This will take time to download and install the first time around, so you should start it as soon as you can.

1. Sun's Java 5 or Java 6. Go to http://java.sun.com and select Downloads -> Java SE. Download the latest Java SE Development Kit (JDK). Select "Download" or "Save", and then double-click the downloaded file to install it. Accept the normal defaults. For Mac OS, you may not need to do this as it already comes with Java pre-installed. For Linux, you may be simply able to use your software installer (e.g., synaptics, apt, yum).
2. Modify the PATH environment variable on your platform so that you can run javac and other java commands from a command window.
   On XP and Vista: Start -> Control Panel . Select Classic View if necessary. Select "System". Click Advanced Settings followed by "Environment Variables". Select the PATH variable in "System Variables" and click on Edit. Add "C:\Program Files\Java\jdk1.X.0_YZ\bin" after a semi-colon (;) to the existing path. Here X, Y, Z correspond to the version of Java you downloaded. Check C:\Program Files\Java directory to check the correct jdk version to use.
3. To test things out so far, open a Command Window or Terminal Window. Type "javac" and hit Return. You should get something like the following in your window.



```
macair:~ aprakash$ javac
Usage: javac <options> <source files>
where possible options include:
  -g                         Generate all debugging info
  -g:none                    Generate no debugging info
  -g:{lines,vars,source}     Generate only some debugging info
  -nowarn                    Generate no warnings
  -verbose                   Output messages about what the compiler is doing
  -deprecation               Output source locations where deprecated APIs are used
  -classpath <path>          Specify where to find user class files
  -cp <path>                 Specify where to find user class files
  -sourcepath <path>         Specify where to find input source files
  -bootclasspath <path>      Override location of bootstrap class files
  -extdirs <dirs>            Override location of installed extensions
  -endorseddirs <dirs>       Override location of endorsed standards path
  -d <directory>             Specify where to place generated class files
  -encoding <encoding>       Specify character encoding used by source files
  -source <release>          Provide source compatibility with specified release
  -target <release>          Generate class files for specific VM version
  -version                   Version information
  -help                      Print a synopsis of standard options
  -X                         Print a synopsis of nonstandard options
  -J<flag>                   Pass <flag> directly to the runtime system
```

4. Create a directory “282” in some standard place (e.g., your Desktop). This is where we will keep our programs.
5. Use Notepad, JEdit, or another editor to create a test Java program. Here is what you should type in:

   ```
   public class MyFirstApp {

       public static void main(String[] args) {
               System.out.println(“Hello World. I <Your name> rules!”);
       }
   }
   ```

   This is the almost the shortest Java program one can write. It simply prints out “Hello World. I <YourName> rule!”.
   Save the program in a file named “MyFirstApp.java” in your 282 directory. Make sure the filename matches the word after “class” (class name). Java requires that.

6. Go to 282 directory in a command window. Type the following to “compile” the program:
   javac MyFirstApp.java

   Hopefully, you do not get any errors. If you do, you need to go back to the editor and make sure you typed the program properly.

   Next, type
   java MyFirstApp

   This should print out “Hello World. I <Your Name> rule!”.

**Part 2: Download a Java Integrated Development Environment (IDE) and run the program using the IDE.**

1. Go to http://www.eclipse.org. Download the latest version of Eclipse. The Download icon is in the middle of the window. Eclipse IDE for Java Developers is likely to be sufficient for now. The Java EE version is larger in size but includes support for developing Java enterprise and web-based applications. Install Eclipse. You need to do this after installing Java.

   Eclipse includes a Java Editor and execution environment. Now try to create a Hello World program using Eclipse. It is a good idea to first create a "Project" (Java Project in case of Eclipse) and then add a file "MyFirstApp.java" to it. Then, edit the file, adding in the same contents.

   Type in and run the same program in Eclipse.

**Part 3:**
Once you have a working program from Part 2, deliberately introduce some errors and see what the compiler reports (use javac to compile the code). The goal is to become familiar with compiler's error messages and be able to map them to the error in your code. The errors you should introduce (one at a time) include:

- Remove one of the opening curly braces
- Remove one of the closing curly braces
- Instead of main, name the function as mian
- Remove the word "static" for the function main
- Remove the word "public" for the function main
- Remove the word "System" in the print statement
- Replace the word "println" with "Println"
- Replace the word "println" with "print". This one is tricky because print is valid, but it does something slightly different from println. You will need to run the program to see the difference.
- Delete the last closing quote in the System.out.println statement
- Delete the closing parenthesis in the System.out.println statement

**What to submit:**

Submit the Java code (.java file) from Part 1 and Part 2 as well screen snapshots (image format) from both parts, showing your program code and its output. For Part 2, make sure you show the program being executed in Eclipse.

Compilers often know that an error occurred, but may go past the point of the error before they realize that something is wrong. They may also have trouble identifying the exact error. For part 3, list all error messages and indicate which ones do not clearly identify the source of the error, i.e., either give an incorrect line number or fail to identify the error clearly. For the println vs. print part, explain the difference in the outputs.

Submit the assignment on ctools.