Object-Oriented Programming

Assignment 1 – Installing Java Software and Running a Program

Due:

Worth: 30 Points

Part 1 (10 points): 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://www.oracle.com/technetwork/java/javase/downloads/index.html and download the Java SE Development Kit (JDK) (Java SE 6 Update 35 is recommended). Select "Download" or "Save", and then double-click the downloaded file to install it. Accept the normal defaults. 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.[1]
3. To test things out so far, open a Command Window or Terminal Window. Type "javac" and hit Return. You should get the details about the command. If you got, 'javac' is not recognized as an internal or external command, so there is something wrong.
4. Use Notepad, 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 JAVA. First program for <Your name> !!");

      }

}
```

[1] Or any other correct installation path.

This is the almost the shortest Java program one can write. Save the program in a file named "MyFirstApp.java". ***Make sure the filename matches the word after "class" (class name). Java requires that.***

5. Go to the directory including the .java file 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
```

```
Note here, you must not include .class extension.
```

This should print out "Hello JAVA. First program for <Your name> !!"

Part 2 (10 points): 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. 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 (10 points):

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.