Tutorial 1

COMP 1406 Winter 2019

Objectives

Using the submission server and basic Java programming: control flow, arrays, output and Strings.

Attendance Quiz

Please log on to cuLearn using one of the computers in the tutorial room and complete the attendance quiz. You can only access the quiz if you log in using one of the computers in the lab. You cannot use a laptop for this. This is a time limited quiz. Be sure to do this as soon as you arrive.

At the end of the tutorial a TA will assign you a grade, call it G, which is 0, 1 or 2, depending on the progress you make during the tutorial. If you spend your time reading mail and chatting with your friends you will receive 0. If you have completed the attendance quiz on time then G will be your tutorial grade. If you have not completed the attendance quiz on time, then your tutorial grade will be max(0, G - 1/2). Each tutorial grade will therefore be one of 0, 0.5, 1.5 or 2.

In order to receive full marks for this tutorial, you must fully complete parts 1 and 2, and make some progress into part 3. **Completing part 2 means that you successfully submit to the server and receive a non-zero grade on it**.

Note that tutorial grades will be finalized two weeks after the tutorial has occurred. Please check your grades on the cuLearn page for your tutorial often to be sure your marks are as expected. We will not be changing marks after two weeks.

It is assumed that you have already read the **SampleJavaCodeProgram.java** file.

0) You will be using a shell to compile and run your java programs. Using Windows **cmd.exe**, create a directory (folder) called **comp1406a0**. First, run the **cmd.exe** program (which we will call the *command line* or *shell) from now on) and enter

```
mkdir comp1406a0
```

Check that the new folder is there by entering

```
dir
```

This will list all files and directories in the current directory that you are in.

Download/copy all the java files for this tutorial into this new directory. To see that they are there, type

```
dir comp1406a0\
```

This lists everything inside the **comp1406a0** directory.

In order compile and run the java programs in this tutorial, you will need to be in the directory that has **comp1406a0** as a sub-directory.

To compile a java file, Tutorial.java for example, you will type

```
javac comp1406a0\Tutorial01.java
```

If the java file compiled without error, you check the comp1406a0 directory again to find a new file called **Tutorial01.class**. This is the java byte code file.

Now, to run the program in the Java Virtual Machine (JVM), type

```
java comp1406a0.Tutorial01
```

Take note that when running the program, we don't have a slash in the command and we don't include the **.class** or **.java** filename extensions.

This is how we will **compile** and **run** all of our Java code this semester.

Now that you know how to compile a Java file, compile and run the **SampleJavaCodeProgram** program. The program will not initially compile. Read the error message and fix the mistake in the code in order to make it compile and run. (Do not spend too much time on fixing it. Google is your friend when you run into compile errors. Check what the error means and see how to fix it. Get help from a TA if you cannot fix this relatively quickly.)

1) Modify the provided **Tutorial01.java** program. The comments in the java file tell you what you should be adding/modifying. Go to the **main** method and start there.

2) Make a zip file of your **comp1406a0** directory. Use the default windows archive program. Right-click on the folder name, chose *send to* and then *compressed (zipped) folder*. Rename your zip file **assignment0-ABCHIJXYZ.zip** (where ABCHIJXYZ is your 9 digit student number). Submit your zip file to the submission server **http://134.117.31.149:9091/**

You can resubmit as many times as needed. Do not move on to the next part until you have a submission that receives a grade greater than 1.

Please note that ALL programming assignments will be submitted in this way. Be sure you understand how to submit to the server.

3) modify the provided **PrintQs.java** program. The program currently asks the user to enter an integer (call it the number N). You will modify the program so that it will draw an NxN grid of Qs. For example, if you enter 7, the program will output

```
QQQQQQQ
QQQQQQQ
QQQQQQQ
QQQQQQQ
QQQQQQQ
QQQQQQQ
QQQQQQQ
```

There are 7 lines printed and each line has 7 Q's printed (without spaces).

Running the program with the value 2 will output

```
QQ
QQ
```

Next, modify your code so that your program displays 4 triangles based on the input N. For example, if the command line argument was 4, then the output should be

```
Q
QQ
QQQ
QQQQ

QQQQ
QQQ
QQ
Q

QQQQ
 QQQ
  QQ
   Q

   Q
  QQ
 QQQ
QQQQ
```

D) Write a program with a static method that computes square roots using the *Babylonian method*.

https://en.wikipedia.org/wiki/Methods_of_computing_square_roots#Babylonian_method

It is an iterative method that keeps making better and better approximations to the square root of a number. The algorithm is terminated when the improvement in successive approximations becomes small. Our version of the Babylonian method to find the square root of a number N is as follows:

1. Let M1 := N/2 be our first guess
2. Let M2 := average of M1 and N/M1
3. If |M1-M2| < epsilon then stop and output M2
4. Otherwise, Let M1 := M2 and go back to step 2

Here, |x| is the absolute value of x and epsilon is a small positive number (like 0.0001). We use := to denote

assignment above.

Your program should prompt the user to enter the the values of N and epsilon. The program should then call your method that prints out all approximations (the value of M2) of the square root of N until is reaches the desired accuracy. When you get the return value you should compare it to the value computed using the square root method given in the **Math** class. You can do other computations (like computing the absolute value) with methods from Math.

https://docs.oracle.com/javase/8/docs/api/java/lang/Math.html