The University of Melbourne
School of Computing and Information Systems

COMP90041
Programming and Software Development
Second Semester, 2018

First Assessed Exercise (lab1)

Submission due Saturday, 18 August 2018, 5:00PM

These exercises are to be assessed, and so **must be done by you alone**. Sophisticated similarity checking software will be used to look for students whose submissions are similar to one another.

See below for instructions on using command line arguments in Java.

1. Write a program `Greetings.java` that expects two string command line arguments, the first a name, and the second a place. The program prints out on one line "`Hello` *name* `from` *place*`.`" and on the next line, "`I'VE ALWAYS WANTED TO GO TO` *PLACE*`.`" where *name* and *place* are the first and second command line arguments, respectively, and *PLACE* is the second printed in all upper case letters. Do not print any extra spaces, even at the beginning or ends of the lines. For example, if the command line arguments are "`Sammy`" and "`Miami`", the program should print out:

   ```
   Hello Sammy from Miami.
   I'VE ALWAYS WANTED TO GO TO MIAMI.
   ```

2. Write a program `Numbers.java` that prints out the sum, difference, product, quotient (integer part) and remainder on division (mod) of the two integer command line arguments, one number per output line. Don't include any extra characters in the output. Do not worry about too many or too few command line arguments, or the command line arguments not being integers. For example, if the command line arguments are 7 and 6, the output should look like:

   ```
   13
   1
   42
   1
   1
   ```

   Hint: You can compute the remainder on division (mod) of `a` by `b` in Java with `a % b`.

   Hint: Remember, you can use `Integer.parseInt(`*s*`)`, where *s* is a string, to turn a string representing an integer into the integer it represents.

Using Command Line Arguments

The command line of a Java program is whatever is placed on the command line after `java` *programname*. Each word or number placed there is made a separate

command line argument when the program is run. If you want multiple words, or text with whitespace or special characters, to be treated as a single argument, you can surround it with single quote characters (`'`). So for example, if your command line is:

```
java Program 42 hello there 'this is a test'
```

when the program is run, it will receive 4 command line arguments: "42", "hello", "there", and "this is a test".

Inside a Java `main` method

```
public static void main(String[] args)
```

the variable `args` holds all the arguments passed to the Java program on the command line as strings. You can access the first command line argument as `args[0]`, the second as `args[1]`, and so on. We will learn why this works in a few weeks, but for now, just use that syntax to access command line arguments.

Submission

You must submit your project from any one of the student unix servers. Make sure the version of your program source files you wish to submit is on these machines (your files are shared between all of them, so any one will do), then `cd` to the directory holding your source code and issue the command:

```
submit COMP90041 lab1 Greetings.java Numbers.java
```

Important: you must wait a minute or two (or more if the servers are busy) after submitting, and then issue the command

```
verify COMP90041 lab1 | less
```

This will show you the test results and the marks from your submission, as well as the file(s) you submitted. If the test results show any problems, correct them and submit again. You may submit as often as you like; only your final submission will be assessed.

If you wish to (re-)submit after the project deadline, you may do so by adding "`.late`" to the end of the project name (*i.e.,* `lab1.late`) in the `submit` and `verify` commands. But note that a penalty, described below, will apply to late submissions, so you should weigh the points you will lose for a late submission against the points you expect to gain by revising your program and submitting again. **It is your responsibility to verify your submission.**

Late Penalties

Late submissions will incur a penalty of 1% of the possible value of that submission per hour late, including evening and weekend hours. This means that a perfect project that is a little more than 2 days late will lose half the marks. These lab exercises are frequent and of low point value, and your lowest lab mark will be dropped. Except in unusual circumstances, I will not grant extensions for lab submissions.

Academic Honesty

This lab submission is part of your final assessment, so cheating is not acceptable. Any form of material exchange between students, whether written, electronic or any other medium, is considered cheating, and so is the soliciting of help from electronic newsgroups. Providing undue assistance is considered as serious as receiving it, and in the case of similarities that indicate exchange of more than basic ideas, formal disciplinary action will be taken for all involved parties.