The University of Melbourne
School of Computing and Information Systems

COMP90041
Programming and Software Development
Second Semester, 2018

Second Assessed Exercise (lab2)

Submission due Saturday, 1 September 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.

1. Write a program `PlaceOrder.java` that prints out a prompt "`Quantity:␣`" (where spaces are shown as ␣), reads in an integer quantity (on the same line), prints out a prompt "`Description:␣`", reads in a description string (which may be many words, up to the end of the input line), prints out a prompt "`Unit␣price:␣`", and reads in a price as a floating point number. Finally it prints out: "`Order␣for␣`q`␣`d" on one line, and "`Total␣price␣`t" on the next, where q is the quantity entered, d is the description entered, but displayed all in upper case, and t is the quantity times the unit price. The total price should be formatted as a floating point number in a field of 14 characters, filled on the left with spaces, and with two digits after the decimal point. For example, one million would be printed as "␣␣␣␣1000000.00". A full example interaction might look like:

   ```
   Quantity:␣ 500
   Description:␣ heads of cauliflower
   Unit␣price:␣ 3.95
   Order␣for␣500␣HEADS␣OF␣CAULIFLOWER
   Total␣price␣␣␣␣␣␣␣1975.00
   ```

   User input is shown in boxes. Note that "`price`" is followed by a space and then the price over 14 characters.

   Important: If you use the `Scanner` class for this question, or any other you submit for assessment, **you must not** create more than one instance of the `Scanner` class. You may use the same instance of `Scanner` as many times as you like.

2. Write a Java program called `Primes.java` that takes two integer arguments on the command line, and prints out the prime numbers from the first command line argument up to the second (inclusive), one number per line. Recall that prime numbers are integers two or more that are only divisible by one and themselves. For example, `java Primes 11 20` should print out (with no spaces or blank lines):

   ```
   11
   13
   17
   19
   ```

Hint: You can check if an integer `a` is divisible by an integer `b` by checking if `a % b` is 0. If it is, then `a` is divisible by `b`.

Hint: A number n is prime if it is not divisible by any number between 2 and $n-1$ inclusive. In fact, it is prime if it is not divisible by any number between 2 and $\sqrt{n}$ (because if $xy = n$, then either $x \leq \sqrt{n}$ or $y \leq \sqrt{n}$). The easiest way to check that $i \leq \sqrt{n}$ is to test if `i*i <= n`.

Hint: Use nested loops: the outer loop ranges from the lower bound to the upper bound (from the command line), checking each number to see if it's prime, and printing it out if it is. The inner loop runs from 2 up to $\sqrt{n}$ (or just $n-1$) to check if n is prime.

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 lab2 PlaceOrder.java Primes.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 lab2 | 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.,* `lab2.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.