Lab Practice Week 1

Students should install either NetBeans and Java SE or, Eclipse and Java SE in your home computer. It is advised to install the same version as those installed in the lab.

All exercises have to be done in the IDE development environment.

What to submit: your solutions to questions 2 and 3 – you may be ask to demo and explain your submission

You should submit your Lab Practice work on a weekly basis to your tutor. Your tutor will expect to see them by your next session. Do not fall behind or leave it to end of the term as the workload will be very heavy towards the end of the term. Zero mark will be given for the week for late submission without acceptable reason.

For external students, instructions will be provided by your tutor on submissions.

Exercises

Before starting the exercises, make sure you have read the relevant materials and all the materials covered in your Lecture notes for Topic 1 (part 1 and part 2).

As many of the exercises involve modifying from previous answers, it is strongly suggested that (in such cases) you copy your project from a previous exercise, then paste and rename it for your current exercise. You can then modify as needed.

1. Download the program *TwoDPs.java* from the unit LMS website and save the file in your working directory. Use the IDE development environment, create and build a project using the source file that you just downloaded. Execute the program.

   Note: You cannot assume that the code provided has no error, as you are expected to debug the code and make sure that it works as expected.

2. Create a new project called *Trunc.java* either by copying and modifying *TwoDPs.java* or by typing that source code from scratch. *Trunc.java* should loop around getting floating point (i.e. type double) numbers and displaying both the number and the running total to the nearest whole number. The program will finish when a number is entered outside of the range -100 to 100. The program should use methods to check whether the number is in this range, and then rounded to the nearest whole number. (NOTE: You are **not** to use the JAVA Round method but write your own method by modifying the *dispTwoDPs* method in *TwoDPs.java*).

3. The following question is adapted from ICT159 and serves as revision. You are to adapt the algorithm that you had learnt in the past, and write the program in JAVA for this exercise.

   Write the JAVA program that reads a string with a maximum length of 30 from the keyboard. The program should then copy the characters from that input string into a second character array (in order of input). However, only if a character is a vowel (a, e, i, o, u) should it be copied into the second array. Once copied, the program should output the values stored in the second array (in order of input). The program should then count and display the number of times each vowel appears in the second array. Also, the program should determine and output the index in the second array where each vowel first appeared or display a suitable message if a particular vowel is not in the array at all.

   Example of input as follows.

   Enter a string to a maximum length of 30:
   I am doing a unit in Murdoch University

   The string is too long. Please enter again.

   Enter a string to a maximum length of 30:
   A unit in Murdoch University

   The output of the second array is:
   auiiuouiei

   The counts are as follows:
   a=1
   e=1
   i=4
   o=1
   u=3

   The index of the second array where each vowel first appeared:
   a = 0
   e = 8
   i = 2
   o = 5
   u = 1

   Another example of input as follows.

   Enter a string to a maximum length of 30:
   aaaaaa

   The output of the second array is:
   aaaaaa

The counts are as follows:
a=6
e=0
i=0
o=0
u=0

The index of the second array where each vowel first appeared:
a = 0
e is not in the input
i is not in the input
o is not in the input
u is not in the input