ICT167 - Laboratory One Notes

Administration

The lecture material covers the conceptual ideas that you are expected to learn in this unit. The tutorial work provides some experience for the practical components of the unit. It is important to work steadily at the tutorials (there is approx. one tutorial per week). This will prove helpful to you when working on the two assignments and for you to understand the concepts covered in the lecture.

This unit uses the Java programming language for tutorial and assignment work. You will need to install an IDE (Eclipse and NetBeans) and Java SE on your working machine. Research on how to install them on your machine. It is advisable to install the same version as those install in your lab. If you are using a different version, you will need to inform your tutor.

Read the file on LMS for further details on the installation. All programs should be developed using this IDE.

The practical components of this unit are the weekly Lab Practice worksheets and two assignments. As outlined in the UILG you are expected to attempt all of the practical components. **You will need to demonstrate and explain your weekly exercises to your tutor/lecturer to get the marks. No LMS or Email submission is allowed for internal students. For external students, you will receive information on submission in an email in Week 1.**

A Lab Practice worksheet will be made available on the unit LMS website for most teaching weeks. Read the relevant lecture material, and think about points that you might want clarification on. Read the Lab Practice exercises and understand what they are asking; think about and plan solutions before asking for help, and in particular before trying to code a solution to a problem. This of course is essential practice for your assignments. You will realise that by developing independent problem solving skill, you can improve your programming skill more effectively. You are advised to plan for your Lab Practice before you go to the class.

Each Lab Practice worksheet has nominated exercises that will be assessed after submission (all assessable questions for each Lab Practice will be assessed at once). Thus, it is important to keep up with the scheduled Lab Practice work; you will find that keeping up with the reading schedule is also advantageous.

It is important for you to explain all the concepts covered in the lecture using your own code provided in the lab exercises and assignments.

Some of the questions in each of the Lab Practice worksheets instruct you to modify a previous question. For assessment, you will need to provide a separate solution for each assessable exercise on each worksheet. That is, assessed exercises will require the source code (from a working project) to be submitted (as requested by your tutor or unit coordinator), for each assessable question for that lab. The simplest way to deal with this is to create a new project for each exercise, and copy the source code of the previous question to the new project and then modify the source for the new project.

For tutorial exercise assessment:

The completed functionality of the required tasks AND the quality of design are assessed. In particular, your solutions should demonstrate good modular design; high cohesion, low coupling, and code re-use. Also, good method and class design (in appropriate exercises). Feedback suggestions for improvement could be provided when your submission is being assessed. Late submission without good reason, and not agreed to by your tutor, will be a zero mark for the Lab Practice for that week.

For assignment assessment:

Both completed functionality AND quality of design are assessed. In particular, your solutions should demonstrate good modular design; high cohesion, low coupling, and code re-use. Also, good method and class design. Feedback suggestions for improvement could be provided when your submission is being assessed. You may be asked for a demonstration/interview session to explain your submission.

Difference between C and Java

1. Terminology - briefly review the concept of a class from your textbook. Note that functions in C are called methods in Java.

2. File/class naming and conventions - class names start with capital letters and use capital letters for subsequently joined words (eg: MyClass). Methods start with lowercase letters but use capital letters for subsequently joined words (eg: addTo-Total()). Also, remember that the class name determines the name of the file it is declared in. A class MyMenuOptions must be in a file named MyMenuOptions.java.

3. Main method - you can have many Java files in a solution (project), but there should be only one class with a main method. This class may be referred to by numerous names; client class, application class, test class, demo class. It is the program that uses other classes, and executes the application.

4. Importing libraries - any library methods can only be used if you import the appropriate class (refer Java API). Eg: java.util.* and java.util.Scanner for I/O.

5. For Input/Output, use the Scanner class methods (for input), and System (for out-put). In Java, System.out is globally available for output, so theoretically System.in (using the Scanner class) can be made globally available for input. Analogy: printf and scanf in C, and cin and cout in C++ are globally available by including the appropriate libraries.

6. Learn from the IDE user guide on their respective websites to perform the tasks.

Exercise Notes

You should learn to write your own notes before going to your tutorial class. This lab notes may not be available for all lab practice exercises.

- Questions 1 and 2. Have a look at dispTwoDPs method in TwoDPs.java; MAKE SURE you can understand HOW it is rounding the floating-point input (to 2 decimal places). Remember what you learned about modular design (particularly high cohesion and low coupling) in ICT159. Note that dispTwoDPs method returns the rounded number, and there is a dedicated output method to display it. If you understand dispTwoDPs, you will find Trunc quite simple. NB: do NOT use the Java 'round' method; modify the method used in dispTwoDPs.

- Question 3. You should have done the algorithm in ICT159 lab exercise. You should understand that the programming concepts and algorithm are independent of the programing language. With the same algorithm, you should be able to code it using different programming languages, as long as you understand the syntax and rules of the programming language. Take note that this program asks you to create a second array, and then do the operation on the second array. Look at the input examples given and use those as a start to test your programs. You should learn the skills in developing good testing examples to test different parts of your program. When developing your testing examples, you should also know what are the expected outputs from your example input directly from your source code (i.e. desk check).