SWE5202 Data Structures and Algorithms

Week 1 worksheet:

**Scheduled Learning Material**

# (to be studied on a "student centred learning" basis during the week)

1. Study the Sun tutorial on Inheritance: <http://java.sun.com/docs/books/tutorial/java/IandI/subclasses.html>
2. Study the material on the differences between association composition and inheritance on the following link ([*http://www.codeproject.com/Articles/22769/Introduction-to-Object-Oriented-*](http://www.codeproject.com/Articles/22769/Introduction-to-Object-Oriented-Programming-Concep#Association)[*Programming-Concep#Association*](http://www.codeproject.com/Articles/22769/Introduction-to-Object-Oriented-Programming-Concep#Association)*)*

You should summarise for yourself the main features of inheritance and the main keywords used to implement inheritance.

# Activity 1 Organising your workspace

* Create a folder in your M drive and call it “**DSAworkspace**”. This folder will be used to store all your java programs. Within this folder, create subfolders for every week (i.e. week1, week2…etc.) to place your programs for every week.

# Activity 2

* **Exercise 1: Aircraft without inheritance**

Start Eclipse and load the programs in the folder ***aircraft-1*** from Moodle. Look at the source code for the Fighter and Bomber classes. How many lines of code are there in these classes?

|  |  |
| --- | --- |
| **Class** | **Number lines in source code** |
| Fighter | 138 |
| Bomber | 137 |
| TOTAL | 275 |

# Note:

For both the Fighter and Bomber classes I have used the Math.random() method to generate random numbers of bullets and bombs. Can you figure out the range of the random numbers generated in both cases? Can you also compare this method of generating random number with the methods available in the Random class?

# The TestAircraft class:

Open the TestAircraft class and look and study carefully how I have written various test methods and how different objects (Fighters and Bomber) are created and added to the relevant ArrayList.

Run the TestAircraft class and observe the output.

# Exercise 2: Aircraft with inheritance

Start Eclipse and load the programs in the folder ***aircraft-2*** from Moodle. Look at the source code for the Fighter, Bomber and Aircraft classes. How many lines of code are there in these classes? Compare with Exercise 1.

|  |  |
| --- | --- |
| **Class** | **Number lines in source code** |
| Fighter | 93 |
| Bomber | 90 |
| Aircraft | 90 |
| TOTAL | 273 |

# Exercise 3: Creating a child class (assessed exercise)

Using the ***aircraft-2*** project, create a new type of aircraft – the TorpedoPlane. This type of aircraft was used to attack shipping by flying towards a ship and dropping a torpedo. The class will have a single field called nbrTorps with suitable accessor and mutator methods. You should also provide two methods the first called dropTorp (which will drop a single torpedo) and toString.

You should look at the other child classes (i.e. Fighter and Bomber) as a guide to doing this exercise.

Write a tester class to create Fighter, Bomber, Torpedoplane objects and test all their methods.

# Activity 3

* Read the file “**notes on inheritance**” from Moodle. Make sure you understand the notes fully.
* Load all programs from the folder “**Inherit**” from Moodle then run them using Eclipse. Try to understand how they work and make notes of the methods used. Write a tester class to test all the methods.

# Activity 4 (assessed exercise)

A class called Vehicle is required by a programmer who is writing software for a car dealer. An object of the class vehicle will consist of a registration number, the make of the Vehicle, the year of manufacture and the current value of the vehicle. The first three of these will need to be set only at the time an object is created. The current value will also be set at the time of creation but may need to be changed during the vehicle’s lifetime.

It will be necessary to have a means of reading the values of all the above data items. A method should also be provided which accepts a year as input and returns the age of the vehicle.

1. Design the Vehicle class using UML notation and then write the code for this class.
2. Design and code a subclass of Vehicle called SecondHandVehicle. The subclass will have an additional attribute, numberOfOwners, which will need to be set at the time a new Vehicle is created and will also need to have read access. An additional method is also required which will report on whether or not the vehicle has had more than one previous owner.
3. Implement the Vehicle and the SecondHandVehicle classes.
4. Write a tester class to test out all the methods of the SecondHandVehicle class.

# You must submit the following to the Moodle “week 1 worksheet submission” link clearly identified with your name and student ID number:

* 1. **A zip folder containing your code for Exercise 3 - Activity 2**

# A zip folder containing your code for Activity 4.

**The deadline for submission is 4th October 2023 @23:55.**