Lab Week12. Linking Classes

Last Week

- Type conversion, casting
- Switch statements
- Enumerated types

Learning Objectives

- Designing and Implementing Multiple Classes for a specific problem
- Using an ArrayList of Objects

Resources

- Lecture Notes - Objects

[Open any portfolio exercises for marking later]

Ex1. In the lecture we looked at design and implementation of a **Student** Class and a **Unit** Class. Open the lecture notes and implement the code we discussed :

- Start a new eclipse project called **Week12**. Add a class called **test**, containing a **static void main**. Add a 2nd class called **Student** (do not include a static void main).
- In class **Student** add members: name, age, mark1, mark2
- Add at least two constructors (1 should allow name, mark1, mark2 to be passed values).
- In your main method, test your Student class. Create 2 Student objects, populate them with values and display them to the console.

- Add a new class to your project called **Unit** with members : name, lecturer, ave_mark1, ave_mark2, ave_unit, class_size and studentList (a list of students as **ArrayList** of Student)
- Add 2 constructors
- Add an **addStudent** method which allows a student to be added to a unit.

```
void addStudent(Student member)
```

Add a **unitAverages** function method which calculates the ave_mark1, ave_mark2 and ave_unit for all students belonging to the unit. It should return the unit average ave_unit.

```
float unitAverages()
```

In your main, create a **Unit** object,
Add two students, with different marks, test **unitAverages** method and display the returned result.

Ex2 Start a new Project called **Week12Music** We will write and **test** some classes as part of a musical album management system for a recording company.
Add a class file called **MusicTest** (with a main method – tick the box),
Add a new class file called **Album** (no main) the class Album should contain the following members

- artist name (or band name)
- album name

- release date (use String)
- number of tracks

Add at least 2 constructors (1 must allow all the members to be passed values).
Add a method to add a track (increment number of tracks)
Add a method to remove a track (decrement number of tracks)

In your main, create 2 different album objects with suitable data to populate the members. Print out ALL the album details to the screen.

Extension exercise 1.: Add a track list member to Album – an ArrayList of String (storing the track title)

Alter your Album methods so that you are able to add a track (including the name), remove a track (by title) and test if a track is contained in an album.

```
ArrayList.add(Object)
ArrayList.remove(Object)
```

see http://beginnersbook.com/2013/12/java-arraylist-add-method-example/ for descriptions and examples of using an ArrayList.

You should design and write these classes (use the lecture notes for reference, where we looked at a similar problem).
You should build a test class that thoroughly tests (black box, see link below) your Class members and methods.
http://softwaretestingfundamentals.com/black-box-testing/

Extension exercise 2. : Add a class track and alter the Album ArrayList to work with this new class instead. Track should contain members

- Track name
- Duration in secs

Alter your Album methods to allow use of this class. Add a function method to calculate and return the Album duration in secs.