WORKSHEET 7a

For this worksheet you are expected to work individually to complete all of the following exercises. The main outcome of this tutorial session is to ensure that you can create a `struct` data-type to build a more complex data-type, also that you can create and use an array of `struct`s, that you can read and write `struct` member variables and use a `union` data-type to provide different storage options.

EXERCISE 1: FINCHES

Difficult Level: ★ **Duration: 20mins**

Create a `struct` called *Measure* with two member variables: feet and inches. Both values should be whole numbers. Write a function that adds two *Measure* `struct` variables together and returns a `struct` with the combined number of feet and inches (i.e. a function with a prototype: `Measure addMeasures(Measure x, Measure y);`). Print the resultant *Measure* (in feet and inches) to the screen.

EXERCISE 2: A WEIGHTY PROBLEM

Difficult Level: ★★ **Duration: 30mins**

Write a program that is able to store details of packages sent by a courier. The courier company requires you to record the weight of the item either in kilograms or in pounds. You should create a new `struct` called *Package* to hold the package information (*ID*, *itemName*, *description*, *weightType* and *weight*) where the *weightType* member variable is an `enum` that defines what form the *weight* is stored in: GRAMS or LBS. You should use a `union` data type to hold this weight value, i.e. use the following two member variables: `int` grams and `float` lbs. Write and test a function that prints the package information in the following format:

```
         ID: 123
  Item Name: Evian 500ml
Description: A 500ml bottle of still Evian water
     Weight: 500g  OR  1.1lbs
```

← output (g or lbs) depends on *weightType*

EXERCISE 3: ARE YOU IN THERE?

Difficult Level: ★★★ **Duration: 1.5hrs**

Write a program to keep a record of all students in a class. To do this you will first need to declare two `struct`s – *Student* and *Class*. The *Student* `struct` should record the following information about each student: *ID*, *fullName* and *finalMark*. The *Class* `struct` should record the following information: 20 students (using an array of *Student* `struct`s), *className* and *classCode*.

Your program should be driven by a menu system with the following menu of options:

```
1 – Set class name and code
2 – Add a new student to the class
3 – Print class details
4 – Find the average class mark
5 – Print student details given ID
6 - Exit
```