Exercise 6 – Interfaces

Objective

The objective of this exercise is to look at object hierarchies and implement some classes which extend from one another and use interfaces to define behaviours.

Overview

This exercise is to extend Exercise 5 (Inheritance) and look at interface which allows the shape to be `Movable`.

Interfaces

We want the circle class to be movable; as we can't inherit from more than one class we use an interface to provide this functionality. Interfaces can be thought of as "jobs the object can also do" and inheritance "what type the object is".

1. First we need to create an interface. Create a new interface called movable
2. Change the class declaration to interface so that it reads

```
public interface Movable {

}
```

3. Declare two methods in the interface, getCurrentLocation() which returns a point object and a move method, which takes in two doubles, an x and a y and returns nothing.
4. In the circle class we want to implement the Movable interface, add this to the class declaration
5. Eclipse will now tell us that we've not implemented all the methods needed to compile the class, implement the getCurrentLocation() and move(x,y) methods in the class.
    a. getCurrentLocation() should just return the current location of the object, this is the same as the centre point for the circle.
    b. The move method should change the current X and Y coordinates of the object by adding the values passed in as the parameters.
        i. Remember to use `super.` to access methods in the super class!
6. Finally we want to test our method, do this by creating a circle object in our main class, calling the move method to change its location and then printing out the contents of the object. Have the X and Y coordinate changed? If not then

remember you can use the debugger to see what is going on as the program runs.

Glossary of key terms

Interface

An interface is a specification of method signatures