Interfaces

COMP2603
Object Oriented Programming 1

Week 5

Abstract Classes

An abstract class is one that cannot be instantiated.

It is useful when we want to factor out common attributes and behaviour from several classes.

Subclasses of an abstract class can then inherit these common attributes and behaviour.

This simplifies the design, development and maintenance of the subclasses.

Outline

- Design Principles
  - Coupling and Cohesion
  - Law of Demeter

Creating an Abstract Class

A class is made abstract using the keyword abstract in the class declaration as follows:

```
public abstract class Sensor{
  public abstract void measure( );
  public abstract boolean testWorking( );
}
```

Subclasses of Abstract Classes

Subclasses would therefore need to provide a method body for all abstract methods as follows

```
public class HumiditySensor extends Sensor{
   public void measure( ){
     //code to measure humidity
   }
   public boolean testWorking(){
     // code to return T/F if the HumiditySensor works
   }
}
```

Interface

An interface makes it possible to deal with objects which are not known except that they implement a minimum set of behaviours specified by the interface.

Example (open and close):

A door

A book

A bank account

Common behaviours that take place in different ways.

Defining an Interface

```
public interface Openable{
  public abstract void open();
  public abstract void close();
}
```

Implementing an Interface

```
public class Door implements Openable{

  // door methods

  public void open(){
    //method implementation
  }
  public void close(){
    //method implementation
  }

}
```

Implementing an Interface

```
public class Account implements Openable{

  // account methods

  public void open(){
    //method implementation
  }
  public void close(){
    //method implementation
  }

}
```

Implementing an Interface

```
public class Book implements Openable{

  // book methods

  public void open(){
    //method implementation
  }
  public void close(){
    //method implementation
  }

}
```

Example

```
Openable[ ] objects = new Openable[3];
objects[0] = new Door();
objects[1] = new Book();
objects[2] = new Account();

for(Openable obj: objects){
    obj.open();
    obj.close();
}
```