Hardik Srivastava

19930306 9

Assignment 5

OOPs

1.
```
import java.util.Scanner;

class Assignment_5_1 {
  public static void main(String[] args){
   Scanner scanner = new Scanner(System.in);
   System.out.println("Enter size of Array:");
   int[] arr = new int[scanner.nextInt()];
   System.out.println(" Enter elements:");
   for( int i =0;i< arr.length ;i++)
      arr[i] = scanner.nextInt();

   arr = sort(arr);
   int largest = arr[0];
   for( int i=0; i<arr.length ; i++){
     if(arr[i] != largest) {
       System.out.println("Second Largest:"+
                                 arr[i]);

       break;

     }
  }

}
private static int[] sort(int[] arr){
   for(int i=0;i<arr.length;i++){
      for(int j = i+1;j< arr.length ;j++){
            if (arr[i] < arr[j ]) {
              int  temp   = arr [i];
               arr[i]  =  arr[j];
               arr[j] = temp;
            }
        }
    }
     return arr;
  }
}
```

```
2  public class Assignment_5_2 {

public static void main (String[] args) {
    try {
        for( int  i = 1 ; i <= 8 ; i ++) {
            checkException(i) ;
        }
    }
    catch (RuntimeException e) {
        System.out.println ("Runtime
                          Exception occured !");
    }
}

private static void checkException (int i)
                    throws RuntimeException {

    try {
        switch (i) {
            case 1 : throw new ArithmeticException();
            case 2 : throw new IOException ();
            case 3 : throw new ArrayIndexOutOfBounds
                                 Exception () ;
            case 4 : throw new StringIndexOutOfBounds
                                 Exception () ;
            case 5 : throw new NullPointerException ();
            case 6 : throw new ClassCastException () ;
            default : throw new RuntimeException();
        }
    }
catch (Exception e){
    System.out.println(e);
}
finally{
    System.out.println("Ended");
}
}
}
```

3. Any class defined inside another class is known as an nested class.

example:

```
class OuterClass {
    class NestedClass {
        // ...
    }
}
```

A nested class has access to the members inc. private members of class in which it is nested.

A non static nested class is an inner class, and it has access to members of the enclosing class (outer class).

example:

```
class CPU {
    double price;
    class Processor {
        double cores;
        string manufacturer;
        double getCache() {
            return 4.3;
        }
    }
//protected nested class
protected class RAM {
    double memory;
    string manufacturer ;
    double getClockSpeed(){
        return 5.5;
    }
  }
}

public class Main {
  public static void main (String[] args){
     //create object of Outer class CPU
     CPU cpu = new CPU();
     // inner class object
     CPU.RAM ram = cpu.new RAM();
  }
}
```

4. Before Java 8, interfaces could have only abstract methods. The implementation of these methods has to be provided in a separate class. So if a new method is to be added in an interface, then its ~~implementation~~ implementation code has to be provided in the class implementing the same interface.

To overcome this, Java 8 introduced the concept of default methods which allow the interfaces to have methods with implementation without affecting the classes that implement the interface

Default methods can be provided to an interface without affecting implementing classes as it includes an implementation. An implementing class can override the default implementation provided by the interface example :

```
interface TestInterface {
    // abstract
    public void square (int a);

    // default
    default void show () {

        System.out.println("Default method
                                  executed");
    }
}
```

Here whichever class implementing this interface would need to compulsarily override the square (int) method whereas show() can be called directly