**January 2013**

**Solutions**

**Q1 (40 marks)**

1. **[10 Marks]**

Write the java code for a Striker class in the Football package. Striker “is a “player. Striker will calculate the number of goal points earned and the number of total points earned for the match, this number will be cumulative.

package football; // .5 Marks

public class Striker extends Player //[1 Mark]

{

private int goalsScored;

private int gamesWon; //[1 Mark]

private int gamesDrawn;

public Striker(String first, String last, int goalsScored, int gamesWon, int gamesDrawn) {

super(first, last); //[1 Mark]

this.goalsScored = goalsScored; //[1.5 Marks]

this.gamesWon = gamesWon;

this.gamesDrawn = gamesDrawn;

}

@Override

public int noGoalPointsEarned() {

return goalsScored \* goalPoints; //[2 Marks]

}

public int totalPointsEarned() //[2 Marks]

{

int totptsEarned = ((super.wins(gamesWon)) + (super.draws(gamesDrawn)) + noGoalPointsEarned());

return totptsEarned;

}

public String toString() //[1 Mark]

{

return "Striker’s name is " + super.toString();

}

}

}

1. **[8 Marks]**

Write the code for a test class, TestFootball. This class should generate two instances of the Striker class.

package football; //.5 Mark

public class TestFootball {

public static void main(String[] args) {

Player S1 = new Striker("john", "doyle", 3, 4, 2); // [.5 Mark]

Player S2 = new Striker("Pat", "Brown", 4, 5, 1); // [.5 Mark]

System.out.println(" " + S1.toString()); // [.5 Mark]

System.out.println("Total goals scored are: " + S1.noGoalPointsEarned()); // [1 Mark]

System.out.println("Total points earned are:" + S1.totalPointsEarned()); // [1 Mark]

if (S1.noGoalPointsEarned() > S2.noGoalPointsEarned()) { // [3 Marks]

System.out.println("Top Goal scorer is: " + S1.firstName + "" + S1.lastName);

} else {

System.out.println("Top Goal scorer is: " + S2.firstName + "" + S2.lastName);

}

}

}

### i) [7 Marks]

Interface FootballPoints  **[1 Mark]**

{

static final int goalPoints = 3; **[3 Marks]**

static final int winPoints = 3;

static final int drawPoints = 1;

int nogoalPointsEarned(); **[3 Marks]**

int highestGoalScorer();

int totalPointsEarned();

}

**ii)**

**[4 Marks]**

Interfaces allow objects to take on many forms. E.g. an Orange has all of the properties of a fruit but it is also a Sphere and has properties relating to that also e.g. diameter etc…

Without the use of interfaces we would be restricted in the description of objects.

1. **[5 Marks]**

The following is an example of a non-static nested class, which, has been added to the Striker’s class. The name of the top scorer should also be displayed.

1. Create an instance of an object for the Striker class **[2 Marks]**

#### Striker ireland = new Striker[“Damien”, “Duff”, 4, 2, 3]

1. Create an instance of an object for the No10Shirt class **[5 Marks]**

Striker.No10Shirt tenshirt = Striker.new No10Shirt[“This is a No. 10 Shirt”);

1. **[5 marks] 2.5 marks each for two of the following**

* Classes that are defined within a single method
* Never declared with an access modifier – they are always restricted to the method in which they are declared
* Hidden from outside world – not even other methods in the same class can access them

**Question 2** **(30 marks)**

1. **(16 marks)**

**public** **class** University{

**private** String uName; //1 mark

**private ArrayList <Department> deptlist = new ArrayList<Department>();** //5 marks

**public** University(String name,**int** deptId[],String deptName[]){ // 2 marks

uName=name;

**for** (**int** i=0;i<4;i++) //3 marks

{

deptlist.add(new Department(deptId[i],deptName[i]));

}

}

**public** **void** showList(){ //5 marks

for (int i=0;i<deptlist.size();i++)

{

System.out.println(deptlist.get(i).getDeptid() +","+

deptlist.get(i).getDeptName());

}

}

}

1. **(9 marks)**

**public** **class** TestUniversity {

**public** **static** **void** main(String[] args){

**int** depids[] = {1001,1002,1003,1004}; //1 mark

String depnames[] = {"Computing", "Business", "Languages", "Engineering"};

//1 mark

University newUniv=**new** University("DCU", depids, depnames); //5 marks

newUniv.showList(); //2 marks

}

}

1. **(5 marks)**

Composition describes the relationship between classes when an object of one class is a data field within another class. The department class which has two member variables representing the name and id number of the department is a data field in the University class as a University is composed of a number of departments.

**Question 3 (30 marks)**

1. **i) 10 Marks – as outlined in the code below:**

**//3 marks for inclusion of the throws at class definition**

public void ValidateDetails()throws TooManyTicketsException,UnknownFilmException

{

**//2 marks for throw new**

if(NoTickets > MaxTicketNum)

throw new TooManyTicketsException("TOO MANY.....");

else

System.out.println("Your booking for "+NoTickets+" tickets is being processed....");

**//2 marks for throw new**

if(FilmName.equals("Pretty Woman")||FilmName.equals("Star Wars"))

System.out.println("You have sucessfully booked " +FilmName+ " for "+NoTickets+" people.

Enjoy the movie.." );

else

throw new UnknownFilmException("WE dont run that show...Sorry");

**//3 marks for correct if structure and code**

}

1. **ii) 15 Marks as outlined in the code below.**

*//5 Marks for try block around the validate details method call*

**try**

**{**

**MyMovie.ValidateDetails();**

**}**

*//5 Marks for the catch TooManyTicketsException()*

**catch(TooManyTicketsException e)**

**{**

**System.out.println("TOO MANY TICKETS HAVE BEEN ORDERED PLEASE**

**RING OUR HOTLINE");**

**}**

*//5 Marks for the catch UnknowFilmException()*

**catch(UnknownFilmException e)**

**{**

**System.out.println("WE DO NOT RUN THAT SHOW............SORRY.");**

**}**

1. **5 marks for explanation containing the following points:**

* The problem with the code is that the catch clauses after the Exception catch will never be executed.
* Only one catch block will be executed and it is always the FIRST matching catch encountered by the program.
* Because Exception is the most general of all catches this one will always be executed.
* Catch clauses should be ordered with the most specific first ending with the most general.

## Question 4 (40 Marks)

1. **(10 marks)**

### Solution -5 marks for explanation and 5 for an accurate example

Unlike non-static inner classes, static inner classes can be instantiated without the necessity of the enclosing class being first instantiated. However they lose the ability to contain nested classes.

1. **(10 marks)**

### Solution -5 marks for variables and 5 marks for methods

Static variables are used to store data that belongs to the class rather than an instance of the class. The value is the same for each instance. They can be referenced without the class being instantiated and by using the name of the class.

Static variables are used to operate on static variables. They can be called without the class being instantiated and by using the name of the class.

1. **(10 marks)**

### Solution -5 marks explanation and 5 for an accurate example

Java provides wrapper classes to manipulate primitive data elements as objects. Such data elements are "wrapped" in an object created around them. Each Java primitive data type has a corresponding wrapper class in the java.lang package. Each wrapper class object encapsulates a single primitive value