

KSU/CCIS/CS	CSC 113	Final exam - Fall 12-13 Time allowed: 3:00
Name: ID:		

EXERCISE 1 (15 marks)

1.1 Write the output of the following program .

```
class A {
    public A() {
        System.out.println(
            "The default constructor of A is invoked");
    }
}

class B extends A {
    public B() {
        System.out.println(
            "The default constructor of B is invoked");
    }
}

public class C {
    public static void main(String[] args) {
        B b = new B();
    }
}
```

Output (6 Marks):

```
The default constructor of A is invoked
The default constructor of B is invoked
```

1.2 Write the output of the following program.

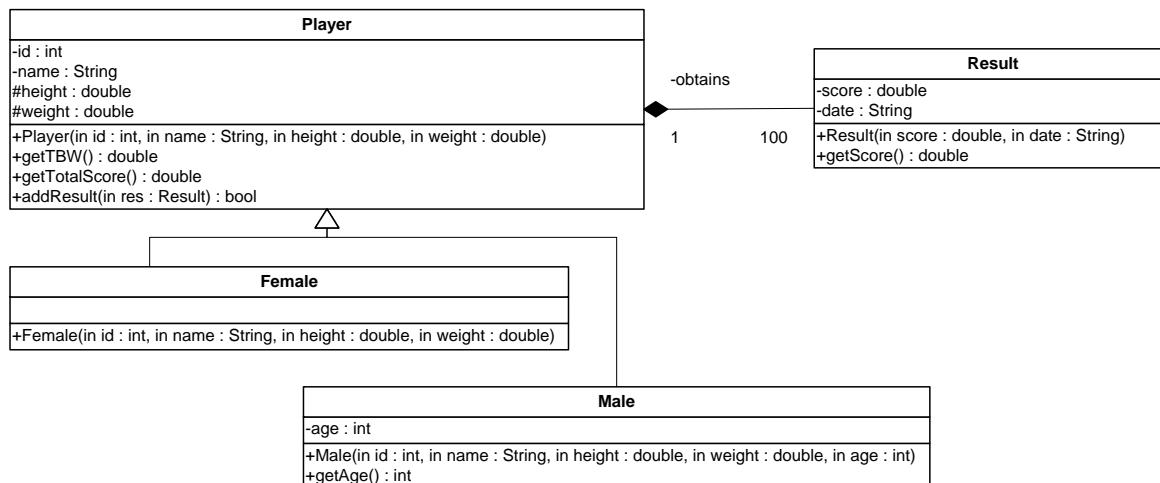
```
class Test {  
    public static void main(String[] args) {  
        try {  
            System.out.println("Welcome to Java");  
            int i = 0;  
            int y = 2/i;  
            System.out.println("Welcome to Java");  
        }  
        catch (ArithmeticException ex) {  
            System.out.println("Welcome to Java");  
        }  
        finally {  
            System.out.println("End of the block");  
        }  
    }  
}
```

Output (9 Marks):

```
Welcome to Java  
Welcome to Java  
End of the block
```

KSU/CCIS/CS	CSC 113	Final exam - Fall 12-13 Time allowed: 3:00
Name: ID:		

EXERCISE 2 (30 marks)



- Class Player**

Player(...): constructor. By default a player may have 100 results.

getTBW(): calculates the Total Body Water (TBW) based on the following formulas:
 For Male: $TBW = 2.447 - (0.09156 \times \text{age}) + (0.1074 \times \text{height}) + (0.3362 \times \text{weight})$
 For Female: $TBW = -2.097 + (0.1069 \times \text{height}) + (0.2466 \times \text{weight})$

getTotalScore(): returns the sum of the scores obtained by the player.

addResult(...): adds a new result to the player results. It returns true if the insertion is done. Otherwise, it returns false.
- Class Result**

Result(...): constructor

getScore(): returns the score of the result.
- Class Male**

Male(...): constructor

getAge(): returns the age of the male.

Write in Java the classes: Player, Result and Male.

Class Player

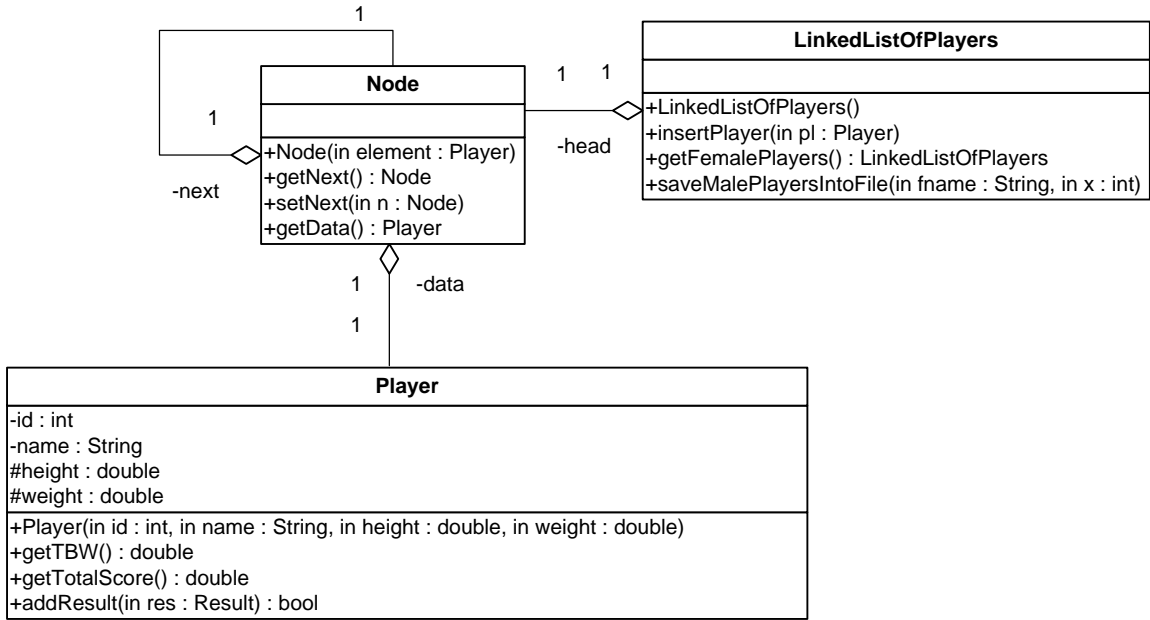
KSU/CCIS/CS	CSC 113	Final exam - Fall 12-13 Time allowed: 3:00
Name: ID:		

Class Result

Class Male

KSU/CCIS/CS	CSC 113	Final exam - Fall 12-13 Time allowed: 3:00
Name: ID:		

EXERCISE 3 (35 marks)



- **Class *LinkedListOfPlayers***

LinkedListOfPlayers(): constructor

insertPlayer(...): inserts a Player object in the list. If the player is a **male**, he will be inserted **at front**. If the player is a **female**, she will be inserted **at back**.

getFemalePlayers(): returns a list of all Female player objects.

saveMalePlayersIntoFile(String fname, int x): writes into the file fname the Male player objects which have an age < x.

Write in Java the class *LinkedListOfPlayers*. Suppose that all the other classes are implemented.

Class LinkedListOfPlayers

(Marks: 16) **public abstract class** Player **implements** java.io.Serializable ... 2

```
{
    private int id;
    private String name;
    protected double height;
    protected double weight;
    private Result[] obtained; ..... 1
    int nb; ..... 1

    public Player(int id, String name, double height, double weight)
    {
        this.id = id;
        this.name = name;
        this.height = height;
        this.weight = weight;
        obtained = new Result[100]; ..... 1
        nb = 0; ..... 1
    }

    public abstract double getTBW(); ..... 2

    public double getTotalScore()
    {
        double sum = 0.0;
        for(int i=0; i<nb; i++) ..... 1
        {
            sum += obtained[i].getScore(); ..... 1
        }
        return sum; ..... 1
    }

    public boolean addResult(Result res)
    {
        if(nb < obtained.length) ..... 1
        {
            obtained[nb] = new Result(res); ..... 2
            nb++; ..... 1
            return true; ..... 0.5
        }
        return false; ..... 0.5
    }
}
```

```

public class Result (Marks: 6)
{
    private double score;
    private String date;

    public Result(double s, String d)
    {
        this.score = s; ..... 1
        this.date = d; ..... 1
    }

    public Result(Result r) ..... 2
    {
        this.score = r.score; ..... 0.5
        this.date = r.date; ..... 0.5
    }

    public double getScore()
    {
        return score; ..... 1
    }
}

```

```

(Marks: 8) public class Male extends Player ..... 1
{
    private int age;

    public Male(int id, String name, double height, double weight, int age)
    {
        super(id, name, height, weight); ..... 2
        this.age = age; ..... 1
    }

    public int getAge()
    {
        return age; ..... 1
    }

    public double getTBW() ..... 2
    {
        return 2.447 - (0.09156 * age) + (0.1074 * height) + (0.3362 *
weight); ..... 1
    }
}

```

```

import java.io.*;

public class LinkedListOfPlayers (Marks: 35)
{
    private Node head; ..... 1

    public LinkedListOfPlayers()
    {
        head = null; ..... 1
    }

    public void insertPlayer(Player pl)
    {
        Node q = new Node(pl); ..... 1

        if ( pl instanceof Male) { ..... 1
            q.setNext(head); ..... 2
            head = q; ..... 1
        }
        else ..... 1
        {
            Node tail = head; ..... 1

            if(head == null) ..... 1
            {
                head = q; ..... 1
            }
            else ..... 1
            {
                while (tail.getNext() != null) ..... 1
                {
                    tail = tail.getNext(); ..... 1
                }
                tail.setNext(q); ..... 2
            }
        }
    }

}

public LinkedListOfPlayers getFemalePlayers()
{
    Node d = head; ..... 1
    LinkedListOfPlayers list = new LinkedListOfPlayers(); ..... 1

    while (d != null) ..... 1
    {
        if (d.getData() instanceof Female) ..... 1
        {
            list.insertPlayer(d.getData()); ..... 1
        }
        d = d.getNext(); ..... 1
    }
    return list; ..... 1
}

```

```

public void saveMalePlayersIntoFile(String fname, int x)
    throws IOException ..... 2
{
    try
    {
        File f = new File(fname); ..... 1
        FileOutputStream fo = new FileOutputStream(f); ..... 1
        ObjectOutputStream os = new ObjectOutputStream(fo); ..... 1

        Node d = head; ..... 1
        while (d != null) ..... 1
        {
            if ( d.getData() instanceof Male &&
                ((Male)d.getData()).getAge() < x) ..... 2
            {
                os.writeObject(d.getData()); ..... 1
            }
            d = d.getNext(); ..... 1
        }
        os.close(); ..... 1
    }
    catch(IOException e)
    {
        System.out.println("Error handling file " + fname);
    }
}
}

```