

CS23333-Object Oriented Programming Using Java-2023

Dashboard / My courses / CS23333-OOPJ-2023 / Lab-07-Interfaces / Lab-07-Logic Building

Quiz navigation



Show one page at a time
Finish review

| | |
|-----------|---------------------------------|
| Status | Finished |
| Started | Monday, 7 October 2024, 5:25 PM |
| Completed | Monday, 7 October 2024, 5:39 PM |
| Duration | 14 mins 22 secs |

Question 1
Correct
Marked out of 5.00
Flag question

Create interfaces shown below.

```
interface Sports {  
    public void setHomeTeam(String name);  
    public void setVisitingTeam(String name);  
}  
  
interface Football extends Sports {  
    public void homeTeamScored(int points);  
    public void visitingTeamScored(int points);  
}
```

create a class College that implements the Football interface and provides the necessary functionality to the abstract methods.

sample Input:

Rajalakshmi
Saveetha
22
21

Output:

Rajalakshmi 22 scored
Saveetha 21 scored
Rajalakshmi is the Winner!

For example:

| Test | Input | Result |
|------|-------------------------------------|---|
| 1 | Rajalakshmi Saveetha 22 21 | Rajalakshmi 22 scored Saveetha 21 scored Rajalakshmi is the winner! |

Answer: (penalty regime: 0 %)

Reset answer

```
1 | import java.util.Scanner;  
2 |  
3 | interface Sports {  
4 |     public void setHomeTeam(String name);  
5 |     public void setVisitingTeam(String name);  
6 | }  
7 |  
8 | interface Football extends Sports {  
9 |     public void homeTeamScored(int points);  
10 |    public void visitingTeamScored(int points);  
11 | }  
12 |  
13 | class College implements Football {  
14 |     String homeTeam;  
15 |     String visitingTeam;  
16 |     public void setHomeTeam(String name) {  
17 |         homeTeam = name;  
18 |     }  
19 |  
20 |     public void setVisitingTeam(String name) {  
21 |         visitingTeam = name;  
22 |     }  
23 |     public void homeTeamScored(int points) {  
24 |         System.out.println(homeTeam + " " + points + " scored");  
25 |     }  
26 |  
27 |     public void visitingTeamScored(int points) {  
28 |         System.out.println(visitingTeam + " " + points + " scored");  
29 |     }  
30 |     public void winningTeam(int homeTeamPoints, int visitingTeamPoints) {  
31 |         if (homeTeamPoints > visitingTeamPoints) {  
32 |             System.out.println(homeTeam + " is the winner!");  
33 |         } else if (homeTeamPoints < visitingTeamPoints) {  
34 |             System.out.println(visitingTeam + " is the winner!");  
35 |         } else {  
36 |             System.out.println("It's a tie match.");  
37 |         }  
38 |     }  
39 | }  
40 |  
41 | public class prog {  
42 |     public static void main(String[] args) {  
43 |         Scanner sc = new Scanner(System.in);  
44 |         String hname = sc.next();  
45 |         String vname = sc.next();  
46 |         int htpoints = sc.nextInt();  
47 |         int vtpoints = sc.nextInt();  
48 |         College s = new College();  
49 |         s.setHomeTeam(hname);  
50 |         s.setVisitingTeam(vname);  
51 |         s.homeTeamScored(htpoints);  
52 |         s.visitingTeamScored(vtpoints);  
53 |     }  
54 | }
```

| | Test | Input | Expected | Got | |
|---|------|-------------------------------------|---|---|---|
| ✓ | 1 | Rajalakshmi Saveetha 22 21 | Rajalakshmi 22 scored Saveetha 21 scored Rajalakshmi is the winner! | Rajalakshmi 22 scored Saveetha 21 scored Rajalakshmi is the winner! | ✓ |
| ✓ | 2 | Anna Balaji 21 21 | Anna 21 scored Balaji 21 scored It's a tie match. | Anna 21 scored Balaji 21 scored It's a tie match. | ✓ |
| ✓ | 3 | SRM VIT 20 21 | SRM 20 scored VIT 21 scored VIT is the winner! | SRM 20 scored VIT 21 scored VIT is the winner! | ✓ |

Passed all tests! ✓

Question 2
Correct
Marked out of 5.00
Flag question

create an interface Playable with a method play() that takes no arguments and returns void. Create three classes Football, Volleyball, and Basketball that implement the Playable interface and override the play() method to play the respective sports.

```
interface Playable {  
    void play();  
}  
  
class Football implements Playable {  
    String name;  
    public Football(String name){  
        this.name=name;  
    }  
    public void play() {  
        System.out.println(name+" is Playing football");  
    }  
}
```

Similarly, create Volleyball and Basketball classes.

Sample output:

Sadhvin is Playing football
Sanjay is Playing volleyball
Sruthi is Playing basketball

For example:

| Test | Input | Result |
|------|---------|------------------------------|
| 1 | Sadhvin | Sadhvin is Playing football |
| | Sanjay | Sanjay is Playing volleyball |
| | Sruthi | Sruthi is Playing basketball |
| 2 | Vijay | Vijay is Playing football |
| | Arun | Arun is Playing volleyball |
| | Balaji | Balaji is Playing basketball |

Answer: (penalty regime: 0 %)

```
1
2 import java.util.*;
3 interface Playable {
4     void play();
5 }
6
7 class Football implements Playable {
8     String name;
9
10    public Football(String name) {
11        this.name = name;
12    }
13
14    @Override
15    public void play() {
16        System.out.println(name + " is Playing football");
17    }
18 }
19
20 class Volleyball implements Playable {
21     String name;
22
23    public Volleyball(String name) {
24        this.name = name;
25    }
26
27    @Override
28    public void play() {
29        System.out.println(name + " is Playing volleyball");
30    }
31 }
32
33 class Basketball implements Playable {
34     String name;
35
36    public Basketball(String name) {
37        this.name = name;
38    }
39
40    @Override
41    public void play() {
42        System.out.println(name + " is Playing basketball");
43    }
44 }
45
46 public class Main {
47     public static void main(String[] args) {
48         Scanner sc=new Scanner(System.in);
49         String s1=sc.nextLine(),s2=sc.nextLine(),s3=sc.nextLine();
50         Football footballPlayer = new Football(s1);
51         Volleyball volleyballPlayer = new Volleyball(s2);
52         Basketball basketballPlayer = new Basketball(s3);
```

| | Test | Input | Expected | Got | |
|---|------|-----------------------------|---|---|---|
| ✓ | 1 | Sadhvin Sanjay Sruthi | Sadhvin is Playing football Sanjay is Playing volleyball Sruthi is Playing basketball | Sadhvin is Playing football Sanjay is Playing volleyball Sruthi is Playing basketball | ✓ |
| ✓ | 2 | Vijay Arun Balaji | Vijay is Playing football Arun is Playing volleyball Balaji is Playing basketball | Vijay is Playing football Arun is Playing volleyball Balaji is Playing basketball | ✓ |

Passed all tests! ✓

Question 3

Correct

Marked out of 5.00

Flag question

RBI issues all national banks to collect interest on all customer loans.

Create an RBI interface with a variable String parentBank="RBI" and abstract method rateOfInterest().

RBI interface has two more methods default and static method.

default void policyNote() {

System.out.println("RBI has a new Policy issued in 2023.");

}

static void regulations(){

System.out.println("RBI has updated new regulations on 2024.");

}

Create two subclasses SBI and Karur which implements the RBI interface.

Provide the necessary code for the abstract method in two sub-classes.

Sample Input/Output:

RBI has a new Policy issued in 2023

RBI has updated new regulations in 2024.

SBI rate of interest: 7.6 per annum.

Karur rate of interest: 7.4 per annum.

For example:

| Test | Result |
|------|---|
| 1 | RBI has a new Policy issued in 2023 RBI has updated new regulations in 2024. SBI rate of interest: 7.6 per annum. Karur rate of interest: 7.4 per annum. |

Answer: (penalty regime: 0 %)

```
1
2 interface RBI {
3     String parentBank = "RBI";
4
5     double rateOfInterest();
6
7     default void policyNote() {
8         System.out.println("RBI has a new Policy issued in 2023");
9     }
10
11     static void regulations() {
12         System.out.println("RBI has updated new regulations in 2024.");
13     }
14 }
15
16 class SBI implements RBI {
17     @Override
18     public double rateOfInterest() {
```

```
22         return 7.6;
23     }
24 }
25
26 class Karur implements RBI {
27     @Override
28     public double rateOfInterest() {
29         return 7.4;
30     }
31 }
32
33 public class Main {
34     public static void main(String[] args) {
35         SBI sbi = new SBI();
36         Karur karur = new Karur();
37
38         sbi.policyNote();
39         RBI.regulations();
40         System.out.println("SBI rate of interest: " + sbi.rateOfInterest() + " per annum.");
41         System.out.println("Karur rate of interest: " + karur.rateOfInterest() + " per annum.");
42     }
43 }
44 }
```

| | Test | Expected | Got | |
|---|------|---|---|---|
| ✓ | 1 | RBI has a new Policy issued in 2023 RBI has updated new regulations in 2024. SBI rate of interest: 7.6 per annum. Karur rate of interest: 7.4 per annum. | RBI has a new Policy issued in 2023 RBI has updated new regulations in 2024. SBI rate of interest: 7.6 per annum. Karur rate of interest: 7.4 per annum. | ✓ |

Passed all tests! ✓

[Finish review](#)

[→ Lab-07-MCQ](#)

Jump to...

0

[Generate series and find Nth element →](#)