***DAY 2***

1. Find out whether the following file will compile. If it does not compile, how you would fix it?

public static void main(String[ ] args) {

int x = 5;

while (x > 1) {

x = x + 1; // x=6

if (x < 3) { // 6<3 condition false. Here we have to decrement the X value less than 3. Then only it will compile.

System.out.println(“small x”); } } }

Ans - small x; (once all int are completed 2^31 then starts from min val -2^31 which is less than 3 so small x gets printed.)

2. Find out whether the following file will compile. If it does not compile, how you would fix it?

class Digit {

public static void main(String[ ] args) {

int x = 1;

while (x < 10) {

if (x > 3) { // 1>3 Here condition is false. So we need to update x value as 4 are more.

System.out.println(“big x”);

} } } }

Ans - no output int x=4; initialize greater number

3. Find out whether the following file will compile. If it does not compile, how you would fix it?

class Loop {

int x = 5;

while (x>1) { // 5>1 Here condition is true.

x = x - 1; // x=5-1=4

if (x < 3) { // 5<3 Here condition is false. Here we need to update x value as 2 or less than.

System.out.println(“small x”); } } }

Ans - small x printed 2 times. (initialize small no. than 3 to x i.e. int x=2)

4. Personalize the Hello World program with your name so that it tells you Hello rather than the somewhat generic "World."

public class HelloName {

public static void main (String [] args){

System.out.println("Hello Saniya !!");

} }

5. Write a program that produces the following output:

public class HelloWorld {

public static void main(String[] args) {

System.out.println("Hello World!\n" + "It's been nice knowing you.\n"+

"Goodbye world! " );

} }

OUTPUT- Hello World!

It's been nice knowing you.

Goodbye world!

6. What is the output of the following program?

class Hexy {

public static void main (String[] args) {

Integer i = 42;

String s = (i<40)?"life": (i>50)?"universe":"everything";

System.out.println(s);

} }

0utput : Compilation fails

A. null

B. life

C. universe

D. everything

E. Compilation fails

F. An exception is thrown at runtime.

Ans - D) Everything

7. Problem Statement:

1. class Example {

2. public static void main(String[] args) {

3. Short s = 15;

4. Boolean b;

5. // b = (s instanceof Short); } }

Which, inserted independently at line 5, will compile? (Choose all that apply.)

A. b = (Number instanceof s); //Number cannot be resolved to a variable

B. b = (s instanceof Short);

C. b = s.instanceof (Short); //"instanceof", Identifier expected

D. b = (s.instanceof Number); //Number cannot be resolved or is not a field

E. b = s.instanceof (Object); // Object can't resolved to var

F. b = (s instanceof String); //String cannot be resolved or is not a field

Ans - B (s instanceOf Short)

8. What is the output of the following program?

class TryIt {

public static void main(String[] args) {

Integer x = 0;

Integer y = 0;

for(Short z = 0; z < 5; z++)

if ((++x > 2) || (++y > 2)) //1>2 || 1>2

x++; // x=2,

System.out.println(x + " " + y);

}

}

Output : X=8, Y=2

9. What is the output of the following program?

class Titanic {

public static void main(String[] args) {

Boolean b1 = true;

Boolean b2 = false;

Boolean b3 = true;

if ((b1 & b2) | (b2 & b3) & b3)

System.out.println("alpha ");

if ((b1 = false) | (b1 & b3) | (b1 | b2))

System.out.println("beta ");

}

}

F F T

F T F

a) beta

b) alpha

c) alpha beta

d) Compilation fails.

e) No output is produced.

f) An exception is thrown at runtime.

Ans - E) no output

10. Given the following program:

class Maybe {

public static void main(String[] args) {

boolean b1 = true;

boolean b2 = false;

System.out.println(!false ^ false);

System.out.println(" " + (!b1 & (b2 = true)));

System.out.println(" " + (b2 ^ b1)); } }

T F F

Which are true?

a) Line 5 produces true.

b) Line 5 produces false.

c) Line 6 produces true.

d) Line 6 produces false.

e) Line 7 produces true.

f) Line 7 produces false.

Ans - A, D, F

***DAY 3***

1. Create a class named Venue with the following member variables / attributes (Default access)

Data Type Variable Name

String name, city

Create another class called Main and write a main method to test the above class.

**Sample Input and Output :**

Enter the venue name - M. A. Chidambaram Stadium Enter the city name - Chennai

Venue Details

Venue Name : M. A. Chidambaram Stadium

City Name : Chennai

**Solution –**

**package** com.javafolder.prg;

**import** java.util.Scanner;

**class** Venue{

String name, city;

}

**public** **class** Main {

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

Scanner sc = **new** Scanner(System.***in***);

System.***out***.print("Enter the venue name - ");

String vname = sc.next();

System.***out***.print("Enter the city name - ");

String vcity = sc.next();

Venue d = **new** Venue ();

d.name= vname;

d.city = vcity;

System.***out***.println("Venue Details \nVenue Name - "+d.name+"\n"+"City Name - "+d.city);

}

}

2. Create a class named Player with the following member variables / attributes (Default access)

Data Type Variable Name

String name, country, skill

Create another class named Main and write a main method to test the above class.

**Sample Input and Output :**

Enter the player name - MS Dhoni Enter the country name - India Enter the skill - All Rounder

Player Details :

Player Name : MS Dhoni Country Name : India Skill : All Rounder

**Solution –**

**class** Player{

String name, country, skill;

}

**public** **class** Main {

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

Scanner sc = **new** Scanner(System.***in***);

System.***out***.print("Enter the player name - ");

String pname = sc.next();

System.***out***.print("Enter the country name - ");

String pcountry = sc.next();

System.***out***.print("Enter the skill - ");

String pskill = sc.next();

Player p = **new** Player ();

p.name = pname;

p.country = pcountry;

p.skill = pskill;

System.***out***.println();

System.***out***.println("Player Details \nPlayer Name - "+p.name+"\n"+"Country Name - "+p.country

+"\n"+"Skill - "+p.skill);

} }

3. Create a class named Delivery with the following public member variables / attributes

Data Type Variable Name

Long over, ball, runs

String batsman, bowler, nonStriker

Include a method in the class named displayDeliveryDetails(). In this method, display the details of the delivery in the format shown in the sample output. This method does not accept any arguments and its return type is void. Create another class called Main and write a main method to test the above class.

**Sample Input and Output :**

Enter the over – 1 Enter the ball - 1 Enter the runs - 4

Enter the batsman name - MS Dhoni

Enter the bowler name - Dale steyn

Enter the nonStriker name - Suresh Raina

Delivery Details :

Over : 1 Ball : 1 Runs : 4

Batsman : MS Dhoni Bowler : Dale steyn NonStriker : Suresh Raina

**Solution –**

**class** Delivery{

**long** over, ball, runs;

String batsman, bowler, nonStriker;

**void** displayDeliveryDetails() {

System.***out***.println("Delivery Details -\nOver - "+ over +"\nBall - "+ball+ "\nRuns - "+ runs

+ "\nBatsman - "+ batsman+ "\nBowler - "+ bowler+ "\nNonStriker - "+ nonStriker);

}

}

**public** **class** Main {

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

Delivery d = **new** Delivery();

Scanner sc = **new** Scanner(System.***in***);

System.***out***.print("Enter the over - ");

**long** dover = sc.nextInt();

System.***out***.print("Enter the balls - ");

**long** dball= sc.nextInt();

System.***out***.print("Enter the runs - ");

**long** druns = sc.nextInt();

sc.nextLine();

System.***out***.print("Enter the Batsman Name - ");

String dbatsman = sc.nextLine();

System.***out***.print("Enter the Bowler Name - ");

String dbowler= sc.nextLine();

System.***out***.print("Enter the Non-Striker Name - ");

String dnonStriker= sc.nextLine();

d.over=dover;

d.ball=dball;

d.runs=druns;

d.batsman=dbatsman;

d.bowler=dbowler;

d.nonStriker=dnonStriker;

d.displayDeliveryDetails();

sc.close();

}

}

4. Create a class named Player with the following member variables / attributes (default access)

Data Type Variable Name

String name, country, skill

Create another class called Main and write a main method to get the player details in a string

seperated by comma. Use String. split() function to display the details.

**Sample Input and Output :**

Enter the player details - MS Dhoni,India,All Rounder

Player Details

Player Name : MS Dhoni Country Name : India Skill : All Rounder

**Solution –**

**class** Player{

String name, country, skill;

}

**public** **class** Main {

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

Scanner sc = **new** Scanner(System.***in***);

System.***out***.print("Enter the Player Details - ");

String input = sc.nextLine();

String str[]=input.split(",");

Player p = **new** Player();

p.name=str[0];

p.country=str[1];

p.skill=str[2];

System.***out***.println("Player Details\n"+ "Player Name - "+p.name + "\nCountry Name - "+p.country + "\nSkill - "+ p.skill);

sc.close();

}

}

5. Create a class named Venue with the following private member variables / attributes

String name ,city

Include appropriate getters and setters.

[Naming Convention: getters : getName getCity setters : setName setCity...]

Create another class and write a main method to test the above class. In the main method, get the choice from the user and update the corresponding venue details.

**Sample Input and Output :**

Enter the venue name - Green Park Stadium Enter the city name - Kanpur

Venue Details

Venue Name : Green Park Stadium

City Name : Kanpur

**Verify and Update Venue Details**

**Menu**

1.Update Venue Name 2.Update City Name 3.All informations Correct/Exit

Type 1 or 2 or 3

**2**

Enter the city name - Chennai

Venue Details

Venue Name : Green Park Stadium City Name : Chennai

**Verify and Update Venue Details**

**Menu**

1.Update Venue Name 2.Update City Name 3.All informations Correct/Exit

Type 1 or 2 or 3

**1**

Enter the venue name - Chidambaram Stadium

Venue Details

Venue Name : Chidambaram Stadium City Name : Chennai

**Verify and Update Venue Details**

**Menu**

1.Update Venue Name 2.Update City Name 3.All informations Correct/Exit

Type 1 or 2 or 3

**3**

Venue Details

Venue Name : Chidambaram Stadium City Name : Chennai

**class** Venue{

String name, city;

**public** String getName() {

**return** name;

}

**public** **void** setName(String name) {

**this**.name = name;

}

**public** String getCity() {

**return** city;

}

**public** **void** setCity(String city) {

**this**.city = city;

}

}

**public** **class** Main{

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

{

Scanner sc = **new** Scanner(System.***in***);

System.***out***.print("Enter venue name - ");

String vname = sc.nextLine();

System.***out***.print("Enter city name - ");

String vcity = sc.nextLine();

Venue v = **new** Venue();

v.name=vname;

v.city=vcity;

System.***out***.println("Venue Details\n"+ "Venue Name - "+vname + "\nCity Name - "+vcity);

System.***out***.println();

System.***out***.println("Verify and Update Venue Details \n"

+ "Menu \n"

+ "1.Update Venue Name \n"

+ "2.Update City Name \r\n"

+ "3.All informations Correct/Exit \r\n"

+ "Type 1 or 2 or 3 ");

**int** no = sc.nextInt();

**switch**(no) {

**case** 1:

System.***out***.print("Enter venue name");

vname = sc.next();

System.***out***.println("Venue Details\n"+ "Venue Name - "+vname + "\nCity Name - "+vcity);

**break**;

**case** 2:

System.***out***.println("Enter city name");

vcity = sc.next();

System.***out***.println("Venue Details\n"+ "Venue Name - "+vname + "\nCity Name - "+vcity);

**break**;

**case** 3:

System.***out***.println("All information correct/exit");

**break**;

**default**:

System.***out***.println("out");

**break**;

}

} }

***DAY 4***

1. Venue Details

[Note : Strictly adhere to the object oriented specifications given as a part of the problem statement.

Use the same class names and member variable names. Follow the naming conventions mentioned for getters / setters. Create 3 separate classes in 3 separate files.]

Create a class named Venue with the following private member variables / attributes

Data Type Variable Name

String name, city

Include appropriate getters, setters and constructors. Naming Convention : getters --- getName,etc... setters --- setName,etc...

Include a default constructor, 2-argument constructor --- the 1st argument corresponds to the name and the 2nd argument corresponds to the city. Override the toString() method to display the venue details in the following format specified in the output.

Create a class named VenueBO and include the following methods

No Method Name Method Description

1 void displayVenueDetails(Venue venue) In this method, display the details of the venue.

Input and Output Format:

Note : The statement " Venue Details" in the output is displayed in the method inside the BO class.

**Sample Input and Output :**

Enter the venue name - M Chidhambaram Stadium

Enter the city name - Chennai

Venue Details

M Chidhambaram Stadium, Chennai

**SOLUTION –**

**Venue Class**

**public** **class** Venuee{

String name, city;

**public** String getName() {

**return** name;

}

**public** **void** setName(String name) {

**this**.name = name;

}

**public** String getCity() {

**return** city;

}

**public** **void** setCity(String city) {

**this**.city = city;

}

**public** Venuee(String name, String city) {

**super**();

**this**.name = name;

**this**.city = city;

}

@Override

**public** String toString() {

**return** "Venuee [name=" + name + ", city=" + city + "]";

} }

**VenueBo class**

**public** **class** VenueBo {

**void** displayVenueDetails(Venuee venue) {

System.***out***.println("Venue Details\n"+ "Venue Name - "+ venue.getName() + "\nCity Name - "+venue.getCity());

} }

**Main class**

**public** **class** Main{

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

Scanner sc = **new** Scanner(System.***in***);

System.***out***.print("Enter venue name - ");

String vname = sc.nextLine();

System.***out***.print("Enter city name - ");

String vcity = sc.nextLine();

Venuee v = **new** Venuee(vname, vcity);

v.setName(vname);

v.setCity(vcity);

VenueBo vb = **new** VenueBo();

vb.displayVenueDetails(v);

}

}

1. Player Details (Array of objects)

Create a class named Player with the following private member variables / attributes

Data Type Variable Name

String name, country, skill

Include appropriate getters, setters and 3-argument constructor - argument corresponds to the value of name, country, skill respectively. Override the toString() method to display the player details in the format specified in the output.

Create a class named PlayerBO and include the following methods

Method Name Method Description

void displayAllPlayerDetails(Player[] playerList) In this method, display the details of all players.

**Input and Output Format:**

Note : The statement " Player Details" in the output is displayed in the method inside the BO class.

Sample Input and Output :

Enter the number of players - 3

Enter the player name - MS Dhoni

Enter the country name - India

Enter the skill - All Rounder

Enter the player name - Suresh Raina

Enter the country name - India

Enter the skill - All Rounder

Enter the player name - Michael Hussey

Enter the country name - Australia

Enter the skill - Batsman

Player Details

MS Dhoni India All Rounder

Suresh Raina India All Rounder

Michael Hussey Australia Batsman

**SOLUTION –**

**Player class**

**public** **class** Player {

**private** String name, country, skill;

**public** String getName() {

**return** name;

}

**public** **void** setName(String name) {

**this**.name = name;

}

**public** String getCountry() {

**return** country;

}

**public** **void** setCountry(String country) {

**this**.country = country;

}

**public** String getSkill() {

**return** skill;

}

**public** **void** setSkill(String skill) {

**this**.skill = skill;

}

**public** Player(String name, String country, String skill) {

**super**();

**this**.name = name;

**this**.country = country;

**this**.skill = skill;

}

@Override

**public** String toString() {

**return** "Player [name=" + name + ", country=" + country + ", skill=" + skill + "]";

} }

**PlayerBo class**

**public** **class** PlayerBo{

**void** displayAllPlayerDetails(Player[] playerList) {

**for**(Player p1:playerList) {

System.***out***.println("Player Details\n"+ "Player Name - "+p1.getName() + "\nCountry Name - "+p1.getCountry() +

"\nSkill - "+ p1.getSkill());

} } }

**Main Class**

**public** **class** Main{

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

Scanner sc = **new** Scanner(System.***in***);

System.***out***.println("Enter the number of player - ");

**int** numOfPlayers = Integer.*parseInt*(sc.nextLine());

Player player[] =**new** Player[numOfPlayers];

**for**(**int** i=0;i<numOfPlayers;i++) {

System.***out***.print("Enter the player name - ");

String pname = sc.next();

System.***out***.print("Enter the country name - ");

String pcountry = sc.next();

System.***out***.print("Enter the skill - ");

String pskill = sc.next();

player[i]=**new** Player(pname,pcountry,pskill);

}

System.***out***.println();

PlayerBo pb = **new** PlayerBo();

pb.displayAllPlayerDetails(player);

} }

1. Method Overriding

Player-International Player. Write a program to illustrate Method Overriding. Create a class named Player with following protected attributes -String name, country. Create appropriate Constructor , method public void displayDetails() to display the player details.

Create a class named InternationalPlayer that extends Player class includes following private attributes

• String capNumber;

• Long noOfTestAppearance, noOfODIAppearance;

Override the method public void displayDetails() defined in the Player class to display InternationalPlayer details and player details. Create a Main class and in the main method test the above class.

**Input and Output Format**

Note: Display the text " Player Details: " inside the method DisplayDetails

**Sample Input and Output 1:**

Enter player name - Virat Kohli

Enter player country - India

Enter the Cap number - 268

Enter the number of test appearance - 48

Enter the number of ODI appearnace - 176

**Player Details:**

Player name : Virat Kohli

Country : India

Cap number : 268

Number of test appearnace : 48

Number of ODI appearnace : 176

**SOLUTION –**

1. Inheritance 1 - Illustration

The task is to calculate the area of the shape using a menu driven application. Create a class called Shape.

Include the following protected data members / attributes:

shapeName – of type String

Include the following methods :

Create a constructor that initializes the shapeName

calculateArea – The return type of this method is Double. This method returns 0.

Create a class called Square that extends Shape

Include the following private data members / attributes:

side – of type Integer.

Include the following methods :

Create a constructor that initializes the side. (1-argument constructor). Initialize the shapeName as "Square".

calculateArea – calculates and returns the area of the square. The return type of this method is Double.

Create a class called Rectangle that extends Shape

Include the following private data members / attributes:

length – of type Integer.

breadth – of type Integer.

Include the following methods :

Create a constructor that initializes the length and breadth. (2-argument constructor). Initialize the shapeName as "Rectangle".

calculateArea - calculates and returns the area of the rectangle. The return type of this method is Double.

Create a class called Circle that extends Shape

Include the following private data members / attributes:

radius – of type Integer.

Include the following methods :

Create a constructor that initializes the radius. (1-argument constructor). Initialize the shapeName as "Circle".

calculateArea – calculates and returns the area of the circle. The return type of this method is Double.

Include appropriate getters and setters.

**Input and Output Format:**

Format the output with two decimal points

Sample Input and Output 1:

1. Rectangle

2. Square

3. Circle

Area Calculator --- Choose your shape

1

Enter length and breadth:

100

40

Area of Rectangle is:4000.00

Sample Input and Output 2:

1. Rectangle

2. Square

3. Circle

Area Calculator --- Choose your shape

2

Enter side:

20

Area of Square is:400.00

Sample Input and Output 3:

1. Rectangle

2. Square

3. Circle

Area Calculator --- Choose your shape

3

Enter Radius:

5

Area of Circle is:78.54

1. Inheritance – Vehicle [Create all classes in separate files]

The task is to get the details of the vehicle and display the details using a menu driven application.

Create a class Vehicle and Include the following protected data members / attributes:

String make, vehicleNumber, fuelType

Int fuelCapacity, cc

Include the following public methods. Create a constructor that initializes all the data members --- public Vehicle(String make,String vehicleNumber,String fuelType,Integer fuelCapacity,Integer cc)

displayMake – Display the make of the vehicle

"displayBasicInfo" – display basic information of the vehicle.

"displayDetailInfo" – An empty method.

Create a class TwoWheeler that extends Vehicle

Include the following private attributes / data members: Boolean kickStartAvailable

Include the following public methods - Include appropriate constructors.

"displayDetailInfo" – displays the availability of kick start.

Create a class FourWheeler that extends Vehicle

Include the following private attributes / data members:

String audioSystem

Int numberOfDoors

Include the following public methods. Include appropriate constructors.

"displayDetailInfo" - displays the audio system and number of doors.

Include getter setters for all the classes.

Create a main class to test the classes defined above.

**Input and Output Format:**

Sample Input Output 1:

1.Four Wheeler

2.Two Wheeler

Enter Vehicle Type: 1

Vehicle Make: Volvo

Vehicle Number: TN01BR9689

Fuel Type: 1.Petrol

2.Diesel: 2

Fuel Capacity: 40

Engine CC: 1960

Audio System: Beats

Number of Doors: 5

\*\*\*Volvo\*\*\*

---Basic Information---

Vehicle Number:TN01BR9689

Fuel Capacity:40

Fuel Type:Diesel

CC:1960

---Detail Information---

Audio System: Beats

Number of Doors: 5

Sample Input and Output 2:

1.Four Wheeler

2.Two Wheeler

Enter Vehicle Type: 2

Vehicle Make: Suzuki

Vehicle Number: TN60Z1234

Fuel Type:

1.Petrol

2.Diesel

1

Fuel Capacity: 15

Engine CC: 150

Kick Start Available(yes/no): yes

\*\*\*Suzuki\*\*\*

---Basic Information---

Vehicle Number:TN60Z1234

Fuel Capacity:15

Fuel Type:Petrol

CC:150

---Detail Information---

Kick Start Available:YES

1. What is the output of the following program?

class Alien {

String invade(short ships) {

return " a few";

}

}

class Defender {

public static void main(String[] args) {

System.out.println(new Alien().invade(7));

}

}

A. many

B. a few

C.  **Compilation fails**

D. The output is not predictable

E. An exception is thrown at runtime

ANS – if System.***out***.println(**new** Alien().invade((**short**) 7)); then ans is A Few

1. What is the output of the following program?

class Fizz {

int x = 5;

public static void main(String[] args) {

final Fizz f1 = new Fizz();

Fizz f2 = new Fizz();

Fizz f3 = FizzSwitch(f1, f2);

System.out.println((f1 == f3) + " " + (f1.x == f3.x));

}

static Fizz FizzSwitch(Fizz x, Fizz y) {

final Fizz z = x;

z.x = 6;

return z;

} }

A. **true true**

B. false true

C. true false

D. false false

E. Compilation fails.

F. An exception is thrown at runtime.

1. What is the output of the following program?

class Knowing {

static final long tooth = 343L;

static long doIt(long tooth) {

System.out.print(++tooth + " ");

return ++tooth;

}

public static void main(String[] args) {

System.out.print(tooth + " ");

final long tooth = 340L;

new Knowing().doIt(tooth);

System.out.println(tooth); } }

A. 343 340 340

B. 343 340 342

C. 343 341 342

**D. 343 341 340**

E. 343 341 343

F. Compilation fails.

G. An exception is thrown at runtime.

1. Which is true? (Choose all that apply.)

A. “X extends Y” is correct if and only if X is a class and Y is an interface.

B. “X extends Y” is correct if and only if X is an interface and Y is a class.

C. “X extends Y” is correct if X and Y are either both classes or both interfaces.

D. “X extends Y” is correct for all combinations of X and Y being classes and/or interfaces.

1. What is the output of the following program?

class Top {

public Top(String s) {

System.out.println("B");

}

}

public class Bottom2 extends Top {

public Bottom2(String s) {

System.out.println("D");

}

public static void main(String[] args) {

new Bottom2("C");

System.out.println(" ");

}

}

}

A. BD

B. DB

C. BDC

D. DBC

E. Compilation fails.

***DAY- 5***

1. Abstract Class I – Shape

Create an abstract class named Shape with the following protected attributes / member variables -

String name

Include a 1-argument constructor, getters and setters, an abstract method named calculateArea() . This method returns a Float value.

Create a class named Circle . The class Circle is a derived class of Shape. Include the following private attributes / member variables - Integer radius

Include a 2-argument constructor. The order of the arguments is name, radius, include getters and setters.

Override the abstract method calculateArea() defined in the Shape class. This method returns the area of the circle. [Take the value of pi as 3.14]

Create a class named Square . The class Square is a derived class of Shape. Include the following private attributes / member variables - Integer side

Include a 2-argument constructor. The order of the arguments is name, side, include getters and setters.

Override the abstract method calculateArea() defined in the Shape class. This method returns the area of the square.

Create a class named Rectangle . The class Rectangle is a derived class of Shape. Include the following private attributes / member variables - Integer length, Integer breadth

Include a 3-argument constructor. The order of the arguments is name, length, breadth, getters and setters.

Override the abstract method calculateArea() defined in the Shape class. This method returns the area of the rectangle.

Create another class called Main. In the method, create instances of the above classes and test the above classes.

**Input and Output Format:**

All Float values are displayed correct to 2 decimal places.

**Sample Input and Output 1:**

Circle

Square

Rectangle

Enter the shape name - Circle

Enter the radius - 25

Area of Circle is 1962.50

**Sample Input and Output 2:**

Circle

Square

Rectangle

Enter the shape name - Square

Enter the radius - 23

Area of Circle is 529.0

**Sample Input and Output 3:**

Circle

Square

Rectangle

Enter the shape name - Rectangle

Enter the length - 45

Enter the breadth - 60

Area of Rectangle is 2700.00

1. Abstract Class II – Card

Create an abstract class named Card with the following protected attributes / member variables.

String holderName, cardNumber, expiryDate;

Include appropriate getters and setters, constructors. In the 3-argument constructor, the order of the arguments is holderName, cardNumber, expiryDate.

Create a class named MembershipCard. The class MembershipCard is a derived class of Card. Include the following private attributes / member variables - Integer rating

Include appropriate getters and setters, appropriate constructors. In the 4-argument constructor, the order of the arguments is holderName, cardNumber, expiryDate, rating.

Create a class named PaybackCard. The class PaybackCard is a derived class of Card. Include the following private attributes / member variables - Integer pointsEarned; Double totalAmount;

Include appropriate getters and setters.

Include appropriate constructors. In the 5-argument constructor, the order of the arguments is holderName, cardNumber, expiryDate, pointsEarned, totalAmount.

Create another class called Main. In the method, create instances of the above classes and test the above classes. The card details are entered separated by a ‘|’.

**Input and Output Format:**

**Sample Input and Output 1:**

Select the Card

1.Payback Card

2.Membership Card

1

Enter the Card Details: Anandhi|12345|14/01/2020

Enter points in card - 1000

Enter Amount - 50000

Anandhi's Payback Card Details: Card Number 12345

Points Earned 1000

Total Amount 50000.0

**Sample Input and Output 2:**

Select the Card

1.Payback Card

2.Membership Card

2

Enter the Card Details: Collin|45678|20/11/2021

Enter rating in card - 10

Collin's Membership Card Details: - Card Number 45678

Rating 10

1. Simple Interface

An Interface consists of a contract of set of methods with only declaration and no implementation. Any class which implements the interface commits that all methods would be implemented. Here is a program to illustrate a simple interface.

1. Create an interface IPlayerStatistics

Add a method with the following prototype

--- public void displayPlayerStatistics

2. Create class Player which implements the IPlayerStatistics Interface

Include private data members :

String name , teamName

Integer noOfMatches , noOfWicketsTaken

Long totalRunsScored

Include an 5 argument constructor with following arguments: name, teamName, noOfMatches, totalRunsScored, noOfWicketsTaken

and implement the interface method public void displayPlayerStatistics to display the player details.

Create a Main class with main method to test test above classes.

**Sample Input and Output 1:**

Enter player name - Ravichandran Ashwin

Enter team name - Chennai Super Kings

Enter number of matches played - 86

Enter total runs scored - 185

Enter number of wickets taken - 89

Player Details

Player name : Ravichandran Ashwin

Team name : Chennai Super Kings

No of matches : 86

Total runsscored : 185

No of wickets taken : 89

4.Interface II

1. Create an interface IPlayerStatistics

Add a method with the following prototype

public void displayPlayerStatistics

2. Create base class CricketPlayer

Include private data members : name, teamName, noOfMatches.

Include an 3 argument constructor with following arguments: name, teamName, noOfMatches.

3. Create derived class Bowler which extends CricketPlayer and implements the interface IplayerStatistics

Include private data members : noOfWickets.

Include an 1 argument constructor with the argument: noOfWickets and implement the interface method public void displayPlayerStatistics to display the player details.

4. Create derived class Batsman which extends CricketPlayer and implements the interface IPlayerStatistics

Include private data members : runs.

Include an 1 argument constructor with the argument: runs and implement the interface method public void displayPlayerStatistics to display the player details.

5. Create derived class WicketKeeper which extends CricketPlayer and implements the interface IPlayerStatistics

Include private data members : noOfCatches, noOfStumpings ,runs, noOfDismissals .

Include an 4 argument constructor with following arguments: noOfCatches, noOfStumpings,runs, noOfDismissals and implement the interface method public void displayPlayerStatistics to display the player details.

6. Create derived class AllRounder which extends CricketPlayer and implements the interface IPlayerStatistics

Include private data members : runs,noOfWickets.

Include an 2 argument constructor with following arguments: runs, noOfWickets and implement the interface method public void displayPlayerStatistics to display the player details.

Create a Main class with main method to test test above classes.

Input and Output Format:

Refer sample input and output for format specifications.

[All text in bold corresponds to input and the rest corresponds to output.]

Sample Input and Output 1:

Menu

1.Bowler

2.Batsman

3.WicketKeeper

4.AllRounder

Enter your choice

1

Enter the Bowler details

Enter player name

Ravichandran Ashwin

Enter team name

Chennai Super Kings

Enter number of matches played

111

Enter number of wickets taken

100

Player name : Ravichandran Ashwin

Team name : Chennai Super Kings

No of matches : 111

No of wickets taken : 100

Do you want to continue?

YES

Menu

1.Bowler

2.Batsman

3.WicketKeeper

4.AllRounder

Enter your choice

4

Enter the AllRounder details

Enter player name

Shane Watson

Enter team name

Royal Challengers Bangalore

Enter number of matches played

94

Enter the runs scored

2551

Enter number of wickets taken

81

Player name : Shane Watson

Team name : Royal Challengers Bangalore

No of matches : 94

Runs scored : 2551

No of wickets taken : 81

Do you want to continue?

NO

Sample Input and output 2:

Menu

1.Bowler

2.Batsman

3.WicketKeeper

4.AllRounder

Enter your choice

2

Enter the Batsman details

Enter player name

Virat Kohli

Enter team name

Royal Challengers Bangalore

Enter number of matches played

139

Enter the runs scored

4110

Player name : Virat Kohli

Team name : Royal Challengers Bangalore

No of matches : 139

Runs scored : 4110

Do you want to continue?

YES

Menu

1.Bowler

2.Batsman

3.WicketKeeper

4.AllRounder

Enter your choice

3

Enter the WicketKeeper details

Enter player name

Mahendra Singh Dhoni

Enter team name

Chennai Super Kings

Enter number of matches played

143

Enter number of catches taken

52

Enter number of stumpings

22

Enter number of dismissals

74

Enter the runs scored

3271

Player name : Mahendra Singh Dhoni

Team name : Chennai Super Kings

No of matches : 143

No of catches taken : 52

No of stumpings : 22

No of dismissals : 74

Runs scored : 3271

Do you want to continue?

NO

Problem Statement:

What is the output of the following program?

**class Eggs {**

**int doX(Long x, Long y) { return 1;}**

**int doX(long... x) { return 2;}**

**int doX(Integer x, Integer y) { return 3; }**

**int doX(Number n, Number m) { return 4; }**

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

**new Eggs().go();**

**}**

**void go() {**

**short s = 7;**

**System.out.println(doX(s,s) + " ");**

**System.out.println(doX(7,7));**

**}**

**}**

A. 1 1

B. 2 1

C. 3 1

D. 4 1

E. 2 3

F. 3 3

G. 4 3

Problem Statement:

What is the output of the following program?

**class Mixer {**

**Mixer() { }**

**Mixer(Mixer m) { m1 = m; }**

**Mixer m1;**

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

**Mixer m2 = new Mixer();**

**Mixer m3 = new Mixer(m2);**

**m3.go();**

**Mixer m4 = m3.m1;**

**m4.go();**

**Mixer m5 = m2.m1;**

**m5.go();**

**}**

**void go() {**

**System.out.println("hi ");**

**}**

**}**

Problem Statement:

What is the output of the following program?

**class Bird {**

**{ System.out.print("b1 "); }**

**public Bird() {**

**System.out.print("b2 ");**

**}**

**}**

**class Raptor extends Bird {**

**static { System.out.print("r1 "); }**

**public Raptor() {**

**System.out.print("r2 ");**

**}**

**{ System.out.print("r3 "); }**

**static { System.out.print("r4 "); }**

**}**

**class Hawk extends Raptor {**

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

**System.out.print("pre ");**

**new Hawk();**

**System.out.println("hawk ");**

**}**

**}**

A. pre b1 b2 r3 r2 hawk

B. pre b2 b1 r2 r3 hawk

C. pre b2 b1 r2 r3 hawk r1 r4

D. r1 r4 pre b1 b2 r3 r2 hawk

E. r1 r4 pre b2 b1 r2 r3 hawk

F. pre r1 r4 b1 b2 r3 r2 hawk

G. pre r1 r4 b2 b1 r2 r3 hawk

H. The order of output cannot be predicted.

I. Compilation fails.

Problem Statement:

What is the output of the following program?

**class Clidders {**

**public final void flipper() {**

**System.out.println("Clidder");**

**}**

**}**

**public class Clidlets extends Clidders {**

**public void flipper() {**

**System.out.println("Flip a Clidlet");**

**super.flipper();**

**}**

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

**new Clidlets().flipper();**

**}**

**}**

A. Flip a Clidlet

B. Flip a Clidder

C. Flip a Clidder

Flip a Clidlet

D. Flip a Clidlet

Flip a Clidder

E. Compilation fails.

Problem Statement:

Given the following:

i. interface Base {

ii. boolean m1 ();

iii. byte m2(short s);

iv. }

Which code fragments will compile? (Choose all that apply.)

a) interface Base2 implements Base { }

b) abstract class Class2 extends Base {

public boolean m1 () { return true; } }

c) abstract class Class2 implements Base { }

d) abstract class Class2 implements Base {

public boolean m1 () { return (true); } }

e) class Class2 implements Base {

boolean m1 () { return false; }

byte m2 (short s) { return 42; } }

Problem Statement:

Which of the following declare a compatible abstract class? (Choose all that apply.)

a) public abstract class Canine { public Bark speak(); }

b) public abstract class Canine { public Bark speak() { } }

c) public class Canine { public abstract Bark speak(); }

d) public class Canine abstract { public abstract Bark speak(); }

Problem Statement:

Given:

public abstract interface Frobnicate { public void twiddle(String s); }

Which is a correct class?

a) **public abstract void twiddle(String s) { }**

**}**

**b) public abstract class Frob implements Frobnicate { }**

**c) public class Frob extends Frobnicate {**

**public void twiddle(Integer i) { }**

**}**

d) **public class Frob implements Frobnicate {**

**public void twiddle(Integer i) { }**

**}**

e) public class Frob implements Frobnicate {

**public void twiddle(String i) { }**

**public void twiddle(Integer s) { }**

**}**

Problem Statement:

Given:

1**. class Zing {**

**2. protected Hmpf h;**

**3. }**

**4. class Woop extends Zing { }**

**5. class Hmpf { }**

Which is true?

A. Woop IS-A Hmpf and HAS-A zing.

B. Zing IS-A Woop and HAS-A Hmpf.

C. Hmpf HAS-A Woop and Woop IS-A Zing.

D. Woop HAS-A Hmpf and Woop IS-A Zing.

E. Zing HAS-A Hmpf and Zing IS-A Woop.

Problem Statement:

Given:

**public class MyOuter {**

**public static class MyInner {**

**public static void foo() ) { }**

**}**

**}**

Which, if placed in a class other than MyOuter or MyInner, instantiates an instance of the nested class?

A. MyOuter.MyInner m = new MyOuter.MyInner();

B. MyOuter.MyInner m2 = new MyInner();

C. MyOuter m = new MyOuter();

MyOuter.MyInner mi = m.new MyOuter.MyInner();

D. MyInner mi = new MyOuter.MyInner();

Problem Statement:

What is the output of the following program?

**public abstract class AbstractTest {**

**public int getNum()) {**

**return 45;**

**}**

**public abstract class Bar {**

**public int getNum()) {**

**return 38;**

**}**

**}**

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

AbstractTest t = **new** AbstractTest()) {

**public** **int** getNum()){

**return 22;**

**}**

**};**

**AbstractTest.Bar f = t.new Bar()) {**

**public int getNum()) {**

**return 57;**

**}**

**};**

**System.out.println()f.getNum()) + " " + t.getNum()));**

**}**

**}**

A. 57 22

B. 45 38

C. 45 57

D. An exception occurs at runtime.

E. Compilation fails.

Handson DAY 7

1. Team Details

Create a class named Team with the following private member variables / attributes

String name;

String coach;

String location;

String players

String captain

Include appropriate getters and setters.

[Naming Convention:

getters : getName getCoach ..

setters : setName, setCoach...]

Include a 5-argument argument constructor in this class. The arguments passed to the constructor are in this order --- name, coach, location,players,captain.

Include a default empty constructor.

Create another class named Main and include a main method to test the above class(Print the output in Main Class).

Input and Output Format:

Refer sample input and output for formatting specifications.

Read the team details as a string value separate by '#'. Use string.split() function to display the team details

All text in bold corresponds to input and the rest corresponds to output.

Sample Input and Output :

Enter the team details

CSK#Stephen Fleming#Chennai#MS Dhoni,Aswin,Raina,Hussey,Maxwel,Bravo,Morkel,Jadeja,Mohit Sharma,Hayden,du plessis,Abhinav Mukund#MS Dhoni

Team : CSK

Coach : Stephen Fleming

Location : Chennai

Players : MS Dhoni,Aswin,Raina,Hussey,Maxwel,Bravo,Morkel,Jadeja,Mohit Sharma,Hayden,du plessis,Abhinav Mukund

Captain : MS Dhoni

2. Wicket details

Create a class named Wicket with the following private member variables / attributes

Long over

Long ball

String wicketType

String playerName

String bowlerName

Include a 5-argument argument constructor in this class. The arguments passed to the constructor are in this order --- over, ball,wicketType,playerName,bowlerName.

Include a default empty constructor.

Include appropriate getters and setters.

[Naming Convention:

getters : getOver getBall ...

setters : setOver, setBall...]

Create another class named Main and include a main method to test the above class(Print the output in Main Class).

Input and Output Format:

Refer sample input and output for formatting specifications.

Use array of objects to read wicket details and use String.split() function to display the wicket details.

All text in bold corresponds to input and the rest corresponds to output.

Sample Input and Output :

Enter the number of wickets

3

Enter the details of wicket 1

5,2,LBW,Gautam Gambir,Aswin

Enter the details of wicket 2

6,6,Bowled,Brad Hogg,Dwayne Bravo

Enter the details of wicket 3

7,3,Stumped,Robin Uthappa,Ravindra Jadeja

Wicket Details

Over : 5

Ball : 2

Wicket Type : LBW

Player Name : Gautam Gambir

Bowler Name : Aswin

Over : 6

Ball : 6

Wicket Type : Bowled

Player Name : Brad Hogg

Bowler Name : Dwayne Bravo

Over : 7

Ball : 3

Wicket Type : Stumped

Player Name : Robin Uthappa

Bowler Name : Ravindra Jadeja

3. IPL Merchandise is an alternative form of revenue and brings substantial income.

Another merchandise company called “Fundas” has come up with a native flair offering customized mugs, team badges, helmets, etc. The company has decided to print a special code on the merchandises. The special code contains the captions of the IPL teams along with the jersey number of the celebrity players of the corresponding team on the merchandises. The code to be printed is considered valid for printing only if their:

--> first word is one of the team's caption [RCB, MI, CSK, SRH, KXIP, DD, KKR, RPSG and GL]

--> second word is the jersey number

Write a program that reads a string S corresponding to the team name and jersey number and validates if the name is eligible to be printed on the stocks.

Include a class UserMainCode with a static method called validateTeam which accepts a string and its return type is bool. In this method display the details as given in sample input and output.

Create a Class Main which would be used to accept two Strings and call the static method called validateTeam present in UserMainCode.

Input Format:

First line of the input is a string S, that corresponds to the team name and jersey number.

Output Format:

Output should display “Valid” if the team name and jersey number is valid for printing. Otherwise print “Invalid”.

Sample Input 1 :

CSK 7

Sample Output 1 :

Valid

Sample Input 2 :

RCB1 18

Sample Output 2 :

Invalid

4. In Cricket it is so common that the players are often called by their last names. Few cricketers have their origins from the same regions of the country which is why they have common last names. Sandeep is now entrusted with yet another task that is to check for the last names of players.

Given two player's names P1 and P2, Sandeep has to write a program to find if the last name of the two given players are the same. Help him do this using String Builder method.

Include a class UserMainCode with a static method called display which accepts two strings and its return type is void. In this method display the details as given in sample input and output.

Create a Class Main which would be used to accepts two Strings and call the static method called display present in UserMainCode.

Input Format:

First line of the input is a string P1, that corresponds to the first player's name.

Second line of the input is a string P2, that corresponds to the second player's name.

Output Format:

Output should print in a single line “Yes” if the last names of the players are the same. Otherwise print “No”.

Sample input

Mohit Sharma

Rohit Sharma

Sample output

Yes

Sample input

Mohit Sharma

Virat Kohli

Sample output

No