6406531929742. ✓ No

Question Number : 134 Question Id : 640653577882 Question Type : SA Calculator : None Response Time : N.A Think Time : N.A Minimum Instruction Time : 0

Correct Marks : 3

Question Label : Short Answer Question

Using the least square approximation, find the length of the projection vector *p* of *b* onto the column space of *A*. Enter the answer correct to two decimal places.

Response Type : Numeric

Evaluation Required For SA : Yes

Show Word Count : Yes

Answers Type : Range

Text Areas : PlainText

Possible Answers :

3.12 to 3.16

Java

Section Id :	64065339074
Section Number :	9
Section type :	Online
Mandatory or Optional :	Mandatory
Number of Questions :	16
Number of Questions to be attempted :	16
Section Marks :	50
Display Number Panel :	Yes
Group All Questions :	No
Enable Mark as Answered Mark for Review and Clear Response :	Yes
Maximum Instruction Time :	0

Sub-Section Number :	1
Sub-Section Id :	64065382602
Question Shuffling Allowed :	No
Is Section Default? :	null

Question Number : 135 Question Id : 640653577888 Question Type : MCQ Is Question Mandatory : No Calculator : None Response Time : N.A Think Time : N.A Minimum Instruction Time : 0

Correct Marks : 0

Question Label : Multiple Choice Question

THIS IS QUESTION PAPER FOR THE SUBJECT "DIPLOMA LEVEL : PROGRAMMING CONCEPTS USING JAVA (COMPUTER BASED EXAM)"

ARE YOU SURE YOU HAVE TO WRITE EXAM FOR THIS SUBJECT?
CROSS CHECK YOUR HALL TICKET TO CONFIRM THE SUBJECTS TO BE WRITTEN.

(IF IT IS NOT THE CORRECT SUBJECT, PLS CHECK THE SECTION AT THE TOP FOR THE SUBJECTS REGISTERED BY YOU)

Options :

6406531929761. ✓ YES

6406531929762. ✗ NO

Sub-Section Number :	2
Sub-Section Id :	64065382603
Question Shuffling Allowed :	Yes
Is Section Default? :	null

Question Number : 136 Question Id : 640653577889 Question Type : MSQ Is Question Mandatory : No Calculator : None Response Time : N.A Think Time : N.A Minimum Instruction Time : 0

Correct Marks : 4 Max. Selectable Options : 0

Question Label : Multiple Select Question

Which among the following statements is/are **true** about activation records?

Options :

6406531929763. ✗ All the variables on the stack (in any activation record) are in their scope.

6406531929764. ✗ Only the variables present in the topmost activation record of the stack are in their lifetime

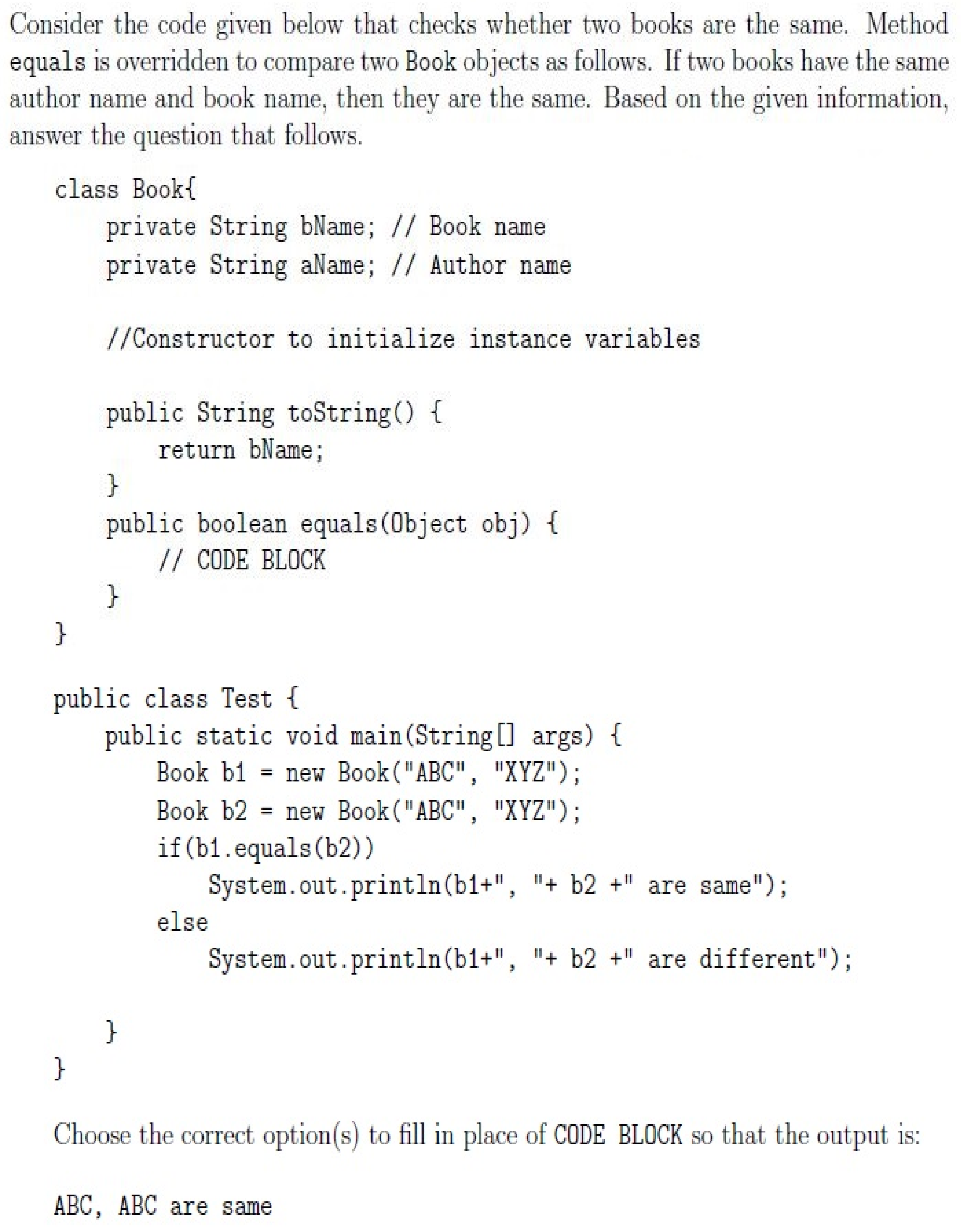
6406531929765. ✓ Every activation record has a control link that points to start of previous record.

6406531929766. ✓ Every activation record has a return value link that points to where to store return value.

Question Number : 137 Question Id : 640653577891 Question Type : MSQ Is Question Mandatory : No Calculator : None Response Time : N.A Think Time : N.A Minimum Instruction Time : 0

Correct Marks : 4 Max. Selectable Options : 0

Question Label : Multiple Select Question



Consider the code given below that checks whether two books are the same. Method equals is overridden to compare two Book objects as follows. If two books have the same author name and book name, then they are the same. Based on the given information, answer the question that follows.

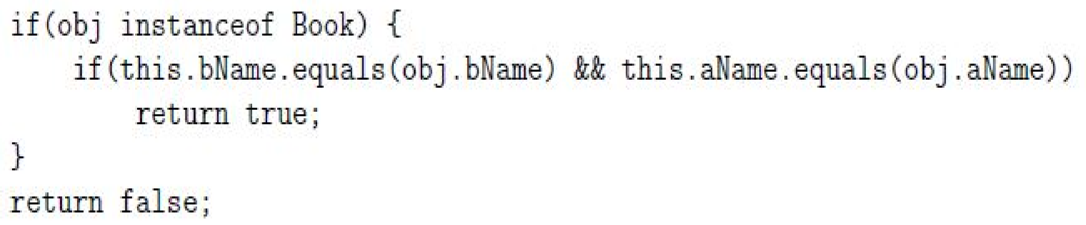
```
class Book{
    private String bName; // Book name
    private String aName; // Author name

    //Constructor to initialize instance variables

    public String toString() {
        return bName;
    }
    public boolean equals(Object obj) {
        // CODE BLOCK
    }
}

public class Test {
    public static void main(String[] args) {
        Book b1 = new Book("ABC", "XYZ");
        Book b2 = new Book("ABC", "XYZ");
        if(b1.equals(b2))
            System.out.println(b1+", "+ b2 +" are same");
        else
            System.out.println(b1+", "+ b2 +" are different");

    }
}
```

Choose the correct option(s) to fill in place of CODE BLOCK so that the output is:

```
ABC, ABC are same
```

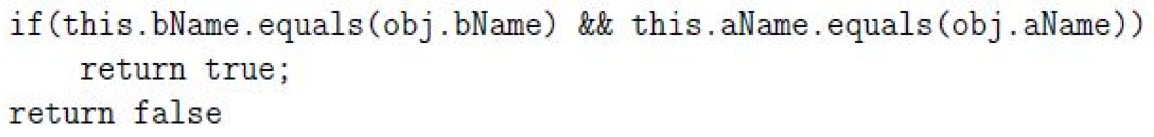
Options :

6406531929771. ✖

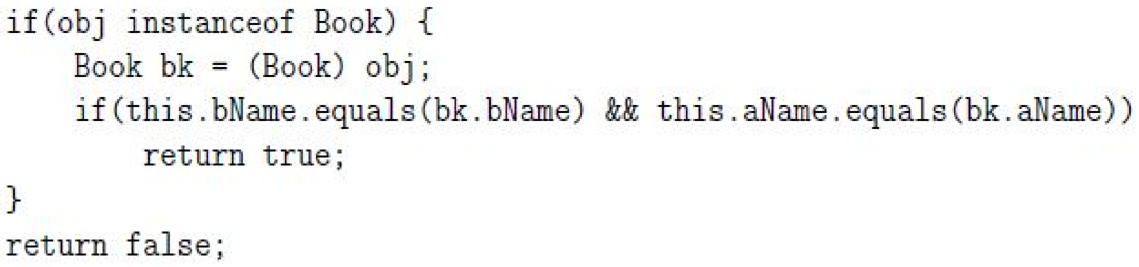


```
if(obj instanceof Book) {
    if(this.bName.equals(obj.bName) && this.aName.equals(obj.aName))
        return true;
}
return false;
```

6406531929772. ✖

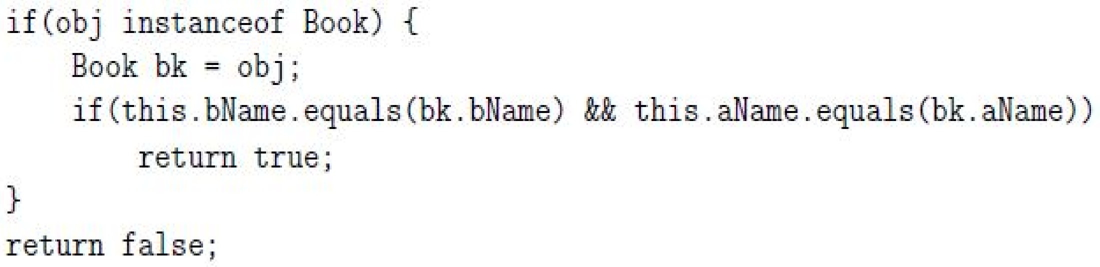


```
if(this.bName.equals(obj.bName) && this.aName.equals(obj.aName))
    return true;
return false
```



```
if(obj instanceof Book) {
    Book bk = (Book) obj;
    if(this.bName.equals(bk.bName) && this.aName.equals(bk.aName))
        return true;
}
return false;
```

6406531929773. ✓



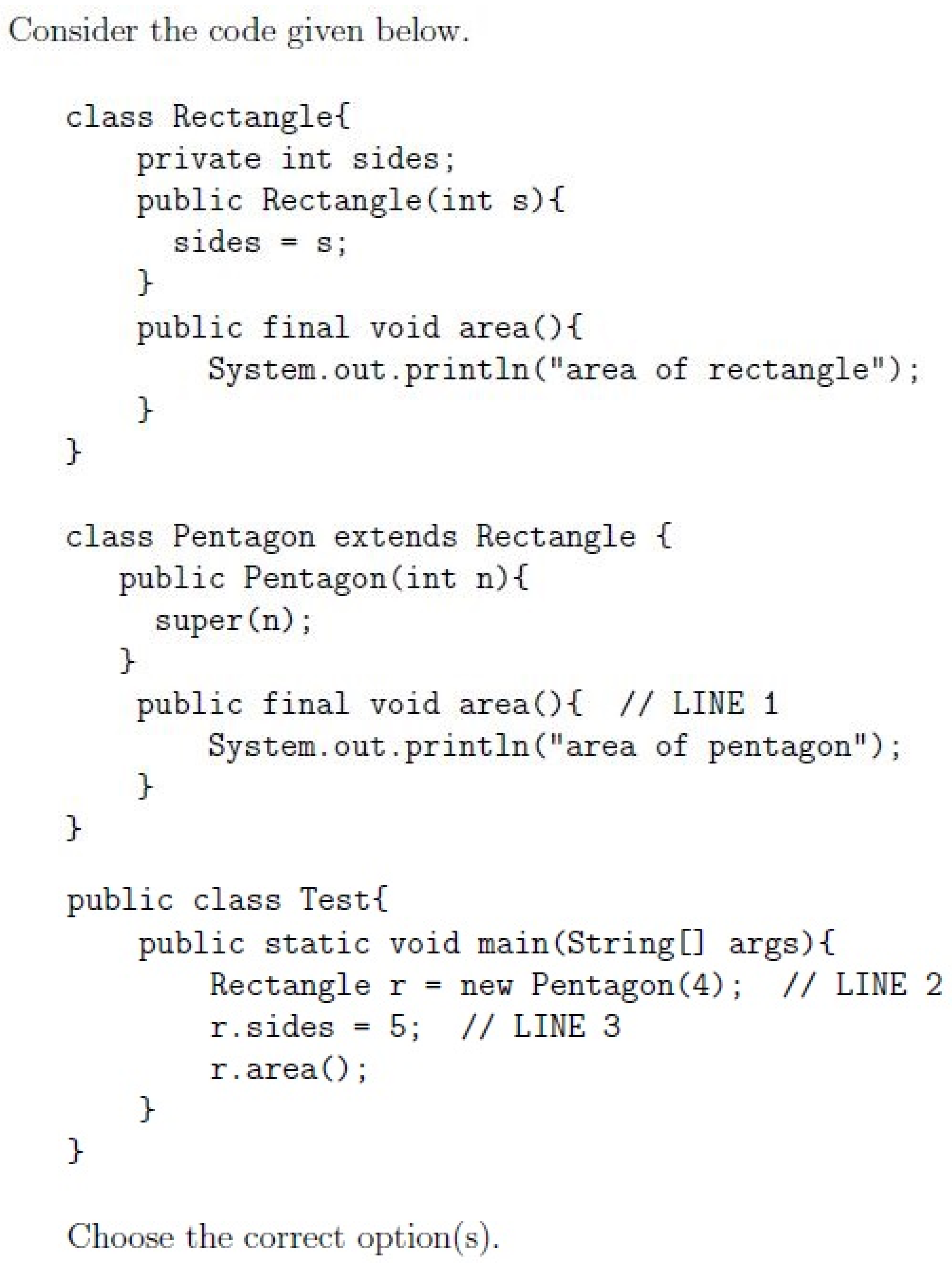
```
if(obj instanceof Book) {
    Book bk = obj;
    if(this.bName.equals(bk.bName) && this.aName.equals(bk.aName))
        return true;
}
return false;
```

6406531929774. ✖

Question Number : 138 Question Id : 640653577894 Question Type : MSQ Is Question Mandatory : No Calculator : None Response Time : N.A Think Time : N.A Minimum Instruction Time : 0

Correct Marks : 4 Max. Selectable Options : 0

Question Label : Multiple Select Question

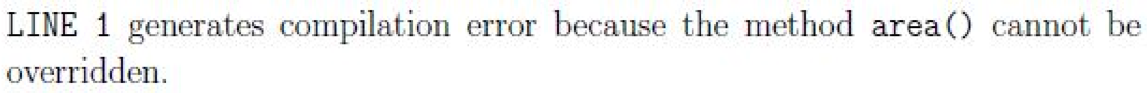


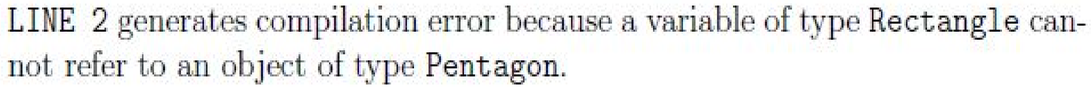
Consider the code given below.

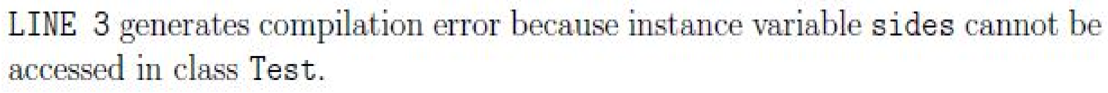
```
class Rectangle{
    private int sides;
    public Rectangle(int s){
      sides = s;
    }
    public final void area(){
        System.out.println("area of rectangle");
    }
}

class Pentagon extends Rectangle {
   public Pentagon(int n){
     super(n);
   }
    public final void area(){  // LINE 1
        System.out.println("area of pentagon");
    }
}

public class Test{
    public static void main(String[] args){
        Rectangle r = new Pentagon(4);  // LINE 2
        r.sides = 5;  // LINE 3
        r.area();
    }
}
```

Choose the correct option(s).

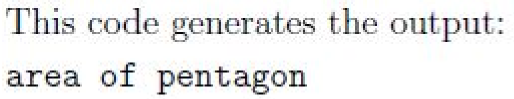
Options :

6406531929783. ✓ 
LINE 1 generates compilation error because the method `area()` cannot be overridden.

6406531929784. ✗ 
LINE 2 generates compilation error because a variable of type `Rectangle` cannot refer to an object of type `Pentagon`.

6406531929785. ✓ 
LINE 3 generates compilation error because instance variable `sides` cannot be accessed in class `Test`.

6406531929786. ✗

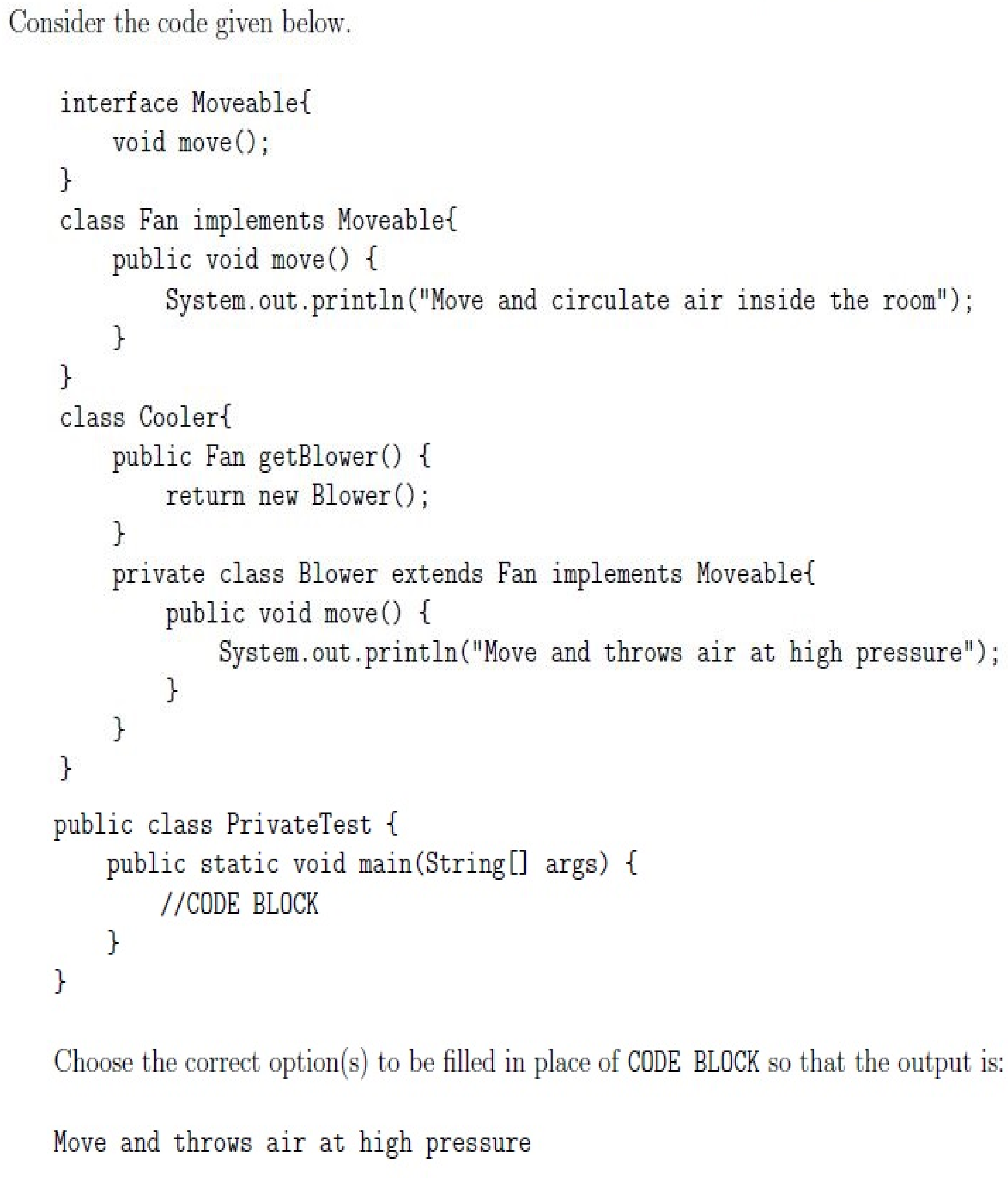

This code generates the output:

```
area of pentagon
```

Question Number : 139 Question Id : 640653577900 Question Type : MSQ Is Question Mandatory : No Calculator : None Response Time : N.A Think Time : N.A Minimum Instruction Time : 0

Correct Marks : 4 Max. Selectable Options : 0

Question Label : Multiple Select Question


Consider the code given below.

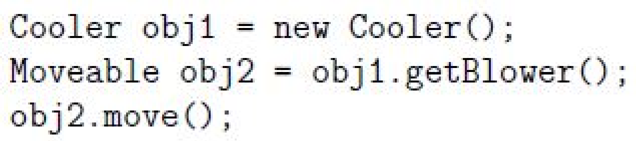
```
interface Moveable{
    void move();
}
class Fan implements Moveable{
    public void move() {
        System.out.println("Move and circulate air inside the room");
    }
}
class Cooler{
    public Fan getBlower() {
        return new Blower();
    }
    private class Blower extends Fan implements Moveable{
        public void move() {
            System.out.println("Move and throws air at high pressure");
        }
    }
}
public class PrivateTest {
    public static void main(String[] args) {
        //CODE BLOCK
    }
}
```

Choose the correct option(s) to be filled in place of CODE BLOCK so that the output is:

```
Move and throws air at high pressure
```

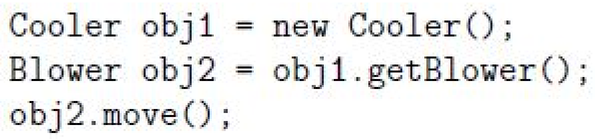
Options :

6406531929807. ✓



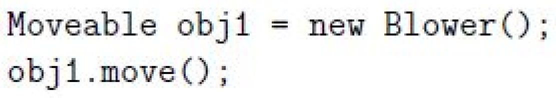
```
Cooler obj1 = new Cooler();
Moveable obj2 = obj1.getBlower();
obj2.move();
```

6406531929808. ✖



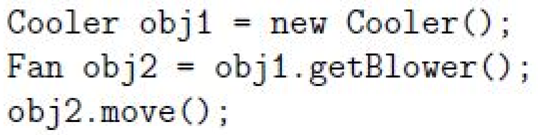
```
Cooler obj1 = new Cooler();
Blower obj2 = obj1.getBlower();
obj2.move();
```

6406531929809. ✖



```
Moveable obj1 = new Blower();
obj1.move();
```

6406531929810. ✓



```
Cooler obj1 = new Cooler();
Fan obj2 = obj1.getBlower();
obj2.move();
```

Sub-Section Number : 3

Sub-Section Id : 64065382604

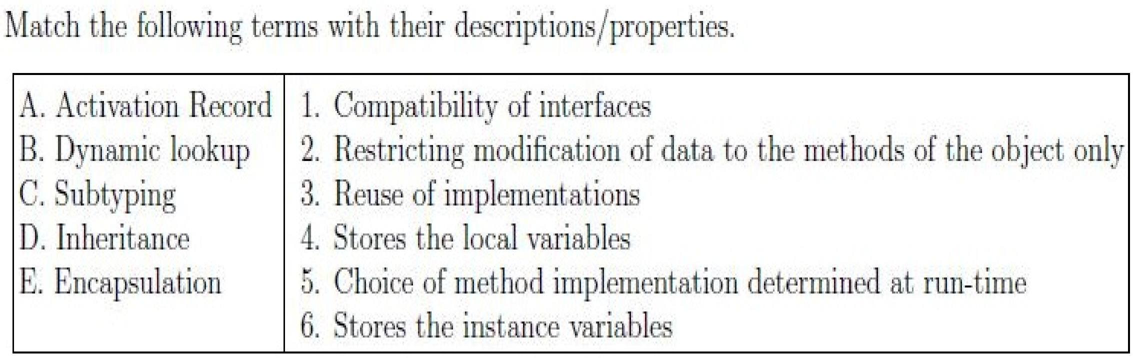
Question Shuffling Allowed : Yes

Is Section Default? : null

Question Number : 140 Question Id : 640653577890 Question Type : MCQ Is Question Mandatory : No Calculator : None Response Time : N.A Think Time : N.A Minimum Instruction Time : 0

Correct Marks : 3

Question Label : Multiple Choice Question



Match the following terms with their descriptions/properties.

A. Activation Record B. Dynamic lookup C. Subtyping D. Inheritance E. Encapsulation	1. Compatibility of interfaces 2. Restricting modification of data to the methods of the object only 3. Reuse of implementations 4. Stores the local variables 5. Choice of method implementation determined at run-time 6. Stores the instance variables

Options :

6406531929767. ✗ A-6, B-5, C-3, D-1, E-2

6406531929768. ✓ A-4, B-5, C-1, D-3, E-2

6406531929769. ✗ A-4, B-2, C-3, D-1, E-5

6406531929770. ✗ A-6, B-2, C-1, D-3, E-5

Question Number : 141 Question Id : 640653577892 Question Type : MCQ Is Question Mandatory : No Calculator : None Response Time : N.A Think Time : N.A Minimum Instruction Time : 0

Correct Marks : 3

Question Label : Multiple Choice Question

Consider the code given below.

```
class Vehicle{
    public void move() {
        System.out.println("Moves");
    }
}
class Bike extends Vehicle{
    public void seating() {
        System.out.println("Bike seating");
    }
}
class Car extends Bike{
    public void seating() {
        System.out.println("Car Seating");
    }
    public void safety() {
        System.out.println("Offers safety");
    }
}
public class Test{
    public static void main(String[] args) {
        Vehicle v = new Bike();
        Bike b = new Car(); // LINE 1
        v.move();
        ((Bike)v).seating(); // LINE 2
        b.safety();  // LINE 3
    }
}
```

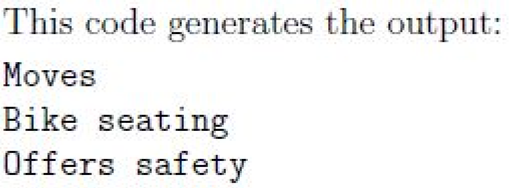
Choose the correct option.

Options :

6406531929775. ✖ LINE 1 generates compilation error because a variable of type Bike cannot refer to an object of type Car.

6406531929776. ✖ LINE 2 generates compilation error because a variable of type Vehicle cannot be type casted to an object of type Bike.

6406531929777. ✔ LINE 3 generates compilation error because the method safety() is not defined in class Bike.

6406531929778. ✖ 

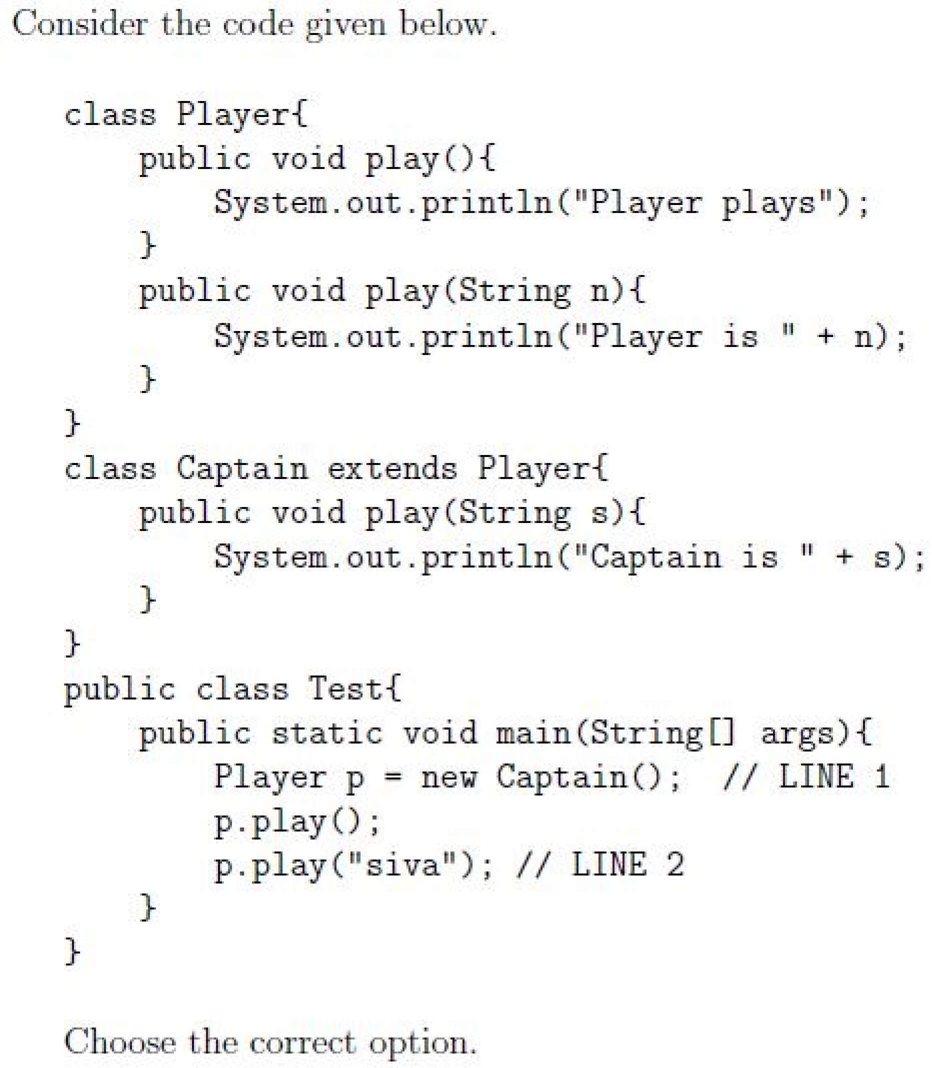
This code generates the output:

```
Moves
Bike seating
Offers safety
```

Question Number : 142 Question Id : 640653577893 Question Type : MCQ Is Question Mandatory : No Calculator : None Response Time : N.A Think Time : N.A Minimum Instruction Time : 0

Correct Marks : 3

Question Label : Multiple Choice Question

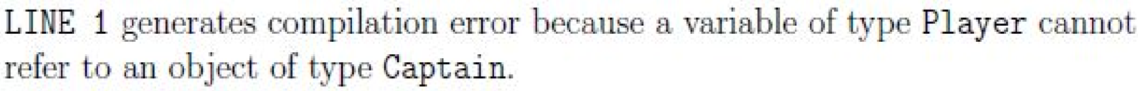


Consider the code given below.

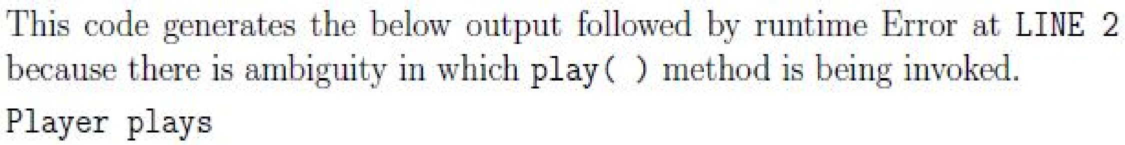
```
class Player{
    public void play(){
        System.out.println("Player plays");
    }
    public void play(String n){
        System.out.println("Player is " + n);
    }
}
class Captain extends Player{
    public void play(String s){
        System.out.println("Captain is " + s);
    }
}
public class Test{
    public static void main(String[] args){
        Player p = new Captain();  // LINE 1
        p.play();
        p.play("siva"); // LINE 2
    }
}
```

Choose the correct option.

Options :

6406531929779. ✖ 

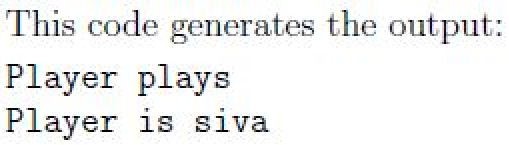
LINE 1 generates compilation error because a variable of type Player cannot refer to an object of type Captain.



This code generates the below output followed by runtime Error at `LINE 2` because there is ambiguity in which `play( )` method is being invoked.

```
Player plays
```

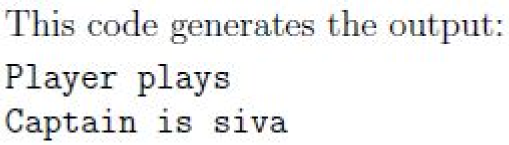
6406531929780. ✖



This code generates the output:

```
Player plays
Player is siva
```

6406531929781. ✖



This code generates the output:

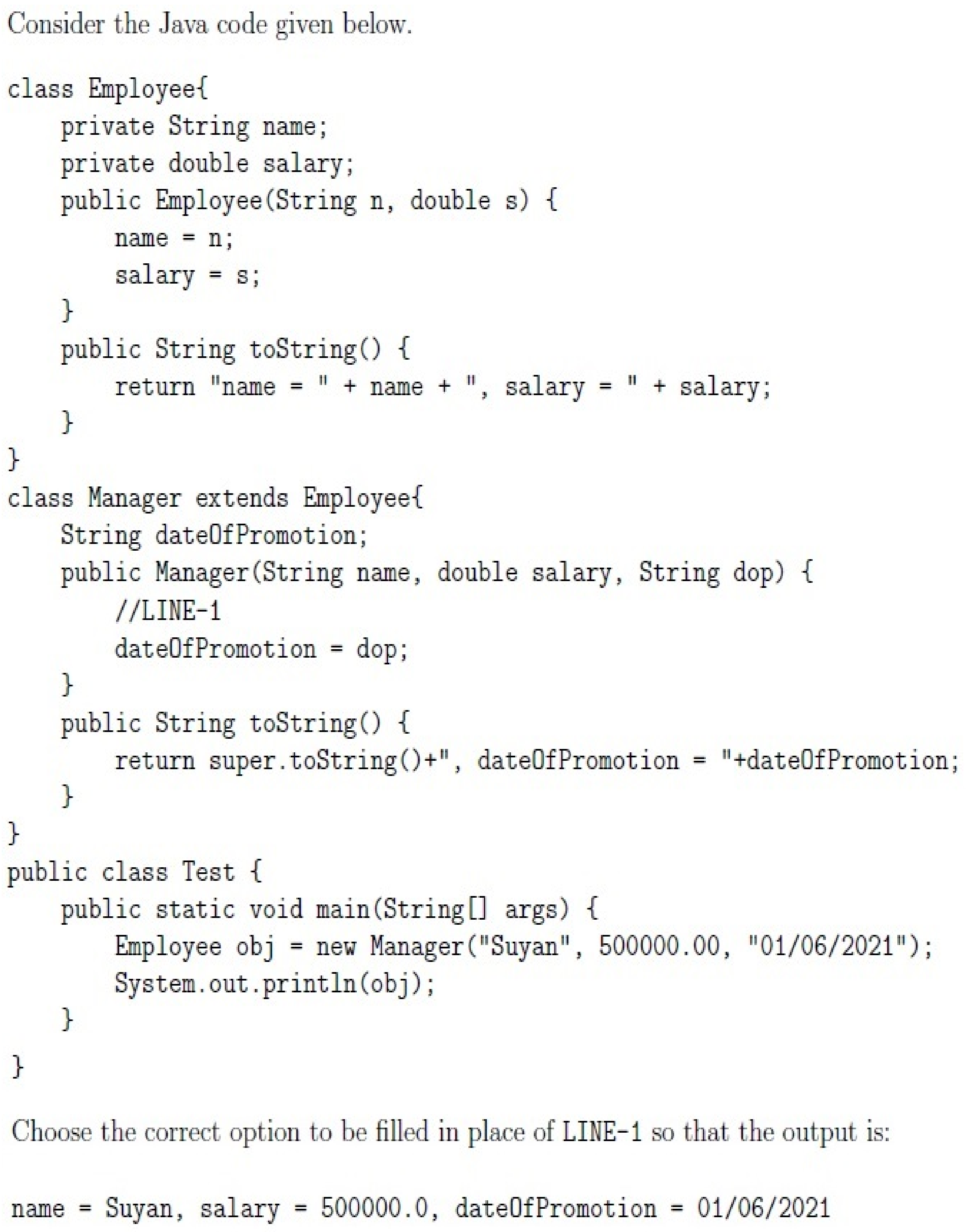
```
Player plays
Captain is siva
```

6406531929782. ✔

Question Number : 143 Question Id : 640653577895 Question Type : MCQ Is Question Mandatory : No Calculator : None Response Time : N.A Think Time : N.A Minimum Instruction Time : 0

Correct Marks : 3

Question Label : Multiple Choice Question



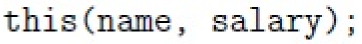
Consider the Java code given below.

```
class Employee{
    private String name;
    private double salary;
    public Employee(String n, double s) {
        name = n;
        salary = s;
    }
    public String toString() {
        return "name = " + name + ", salary = " + salary;
    }
}
class Manager extends Employee{
    String dateOfPromotion;
    public Manager(String name, double salary, String dop) {
        //LINE-1
        dateOfPromotion = dop;
    }
    public String toString() {
        return super.toString()+", dateOfPromotion = "+dateOfPromotion;
    }
}
public class Test {
    public static void main(String[] args) {
        Employee obj = new Manager("Suyan", 500000.00, "01/06/2021");
        System.out.println(obj);
    }
}
```

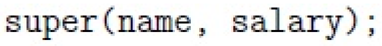
Choose the correct option to be filled in place of LINE-1 so that the output is:

```
name = Suyan, salary = 500000.0, dateOfPromotion = 01/06/2021
```

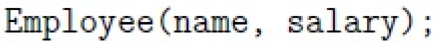
Options :

6406531929787. ✗ 

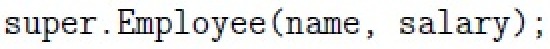
```
this(name, salary);
```

6406531929788. ✓ 

```
super(name, salary);
```

6406531929789. ✗ 

```
Employee(name, salary);
```

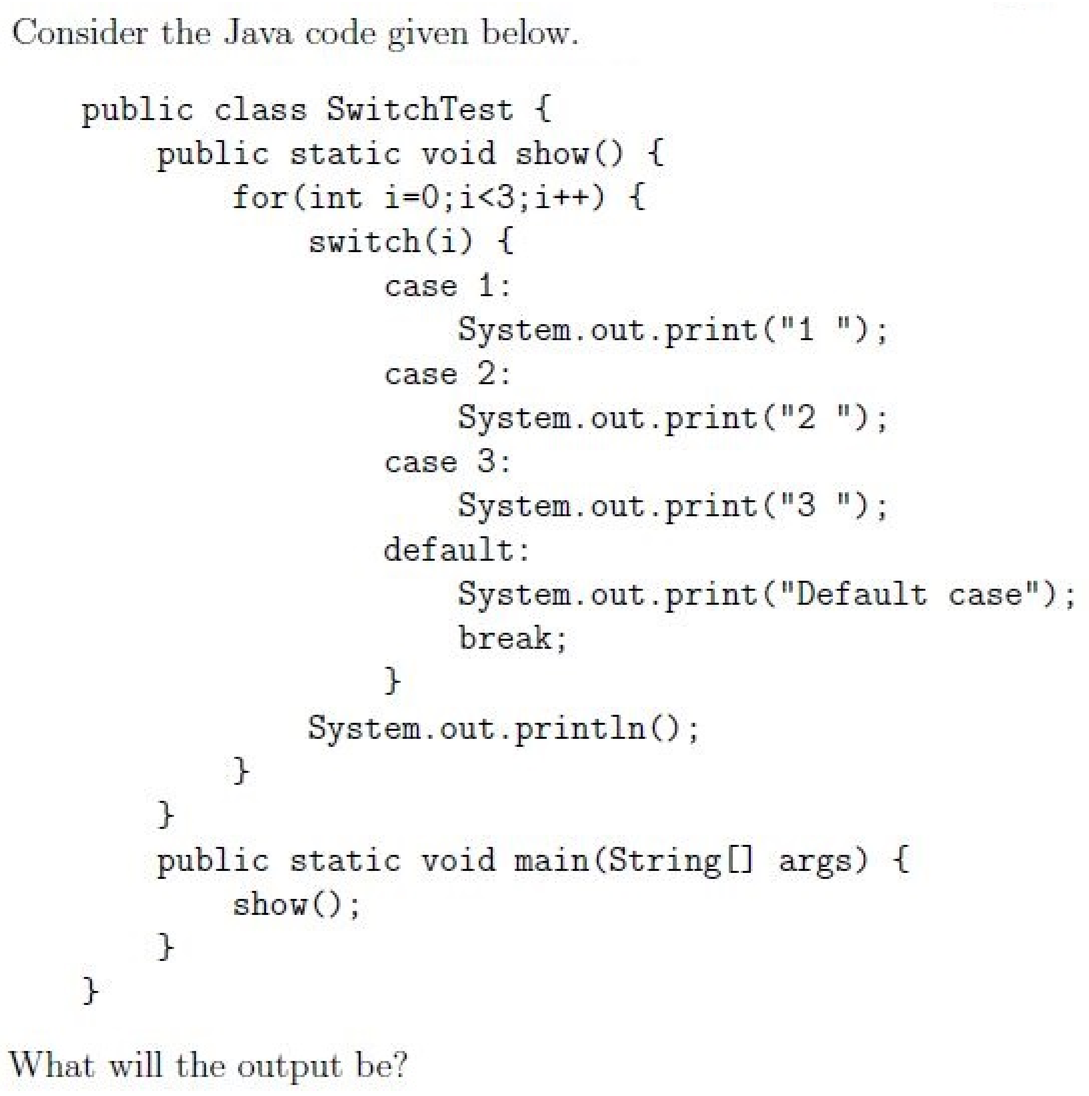
6406531929790. ✗ 

```
super.Employee(name, salary);
```

Question Number : 144 Question Id : 640653577896 Question Type : MCQ Is Question Mandatory : No Calculator : None Response Time : N.A Think Time : N.A Minimum Instruction Time : 0

Correct Marks : 3

Question Label : Multiple Choice Question



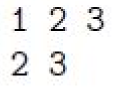
Consider the Java code given below.

```
public class SwitchTest {
    public static void show() {
        for(int i=0;i<3;i++) {
            switch(i) {
                case 1:
                    System.out.print("1 ");
                case 2:
                    System.out.print("2 ");
                case 3:
                    System.out.print("3 ");
                default:
                    System.out.print("Default case");
                    break;
                }
            System.out.println();
        }
    }
    public static void main(String[] args) {
        show();
    }
}
```

What will the output be?

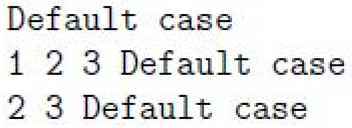
Options :

6406531929791. ✖

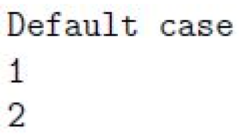


```
1 2 3
2 3
```

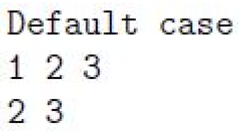
6406531929792. ✔



```
Default case
1 2 3 Default case
2 3 Default case
```

6406531929793. ✖ 

```
Default case
1
2
```

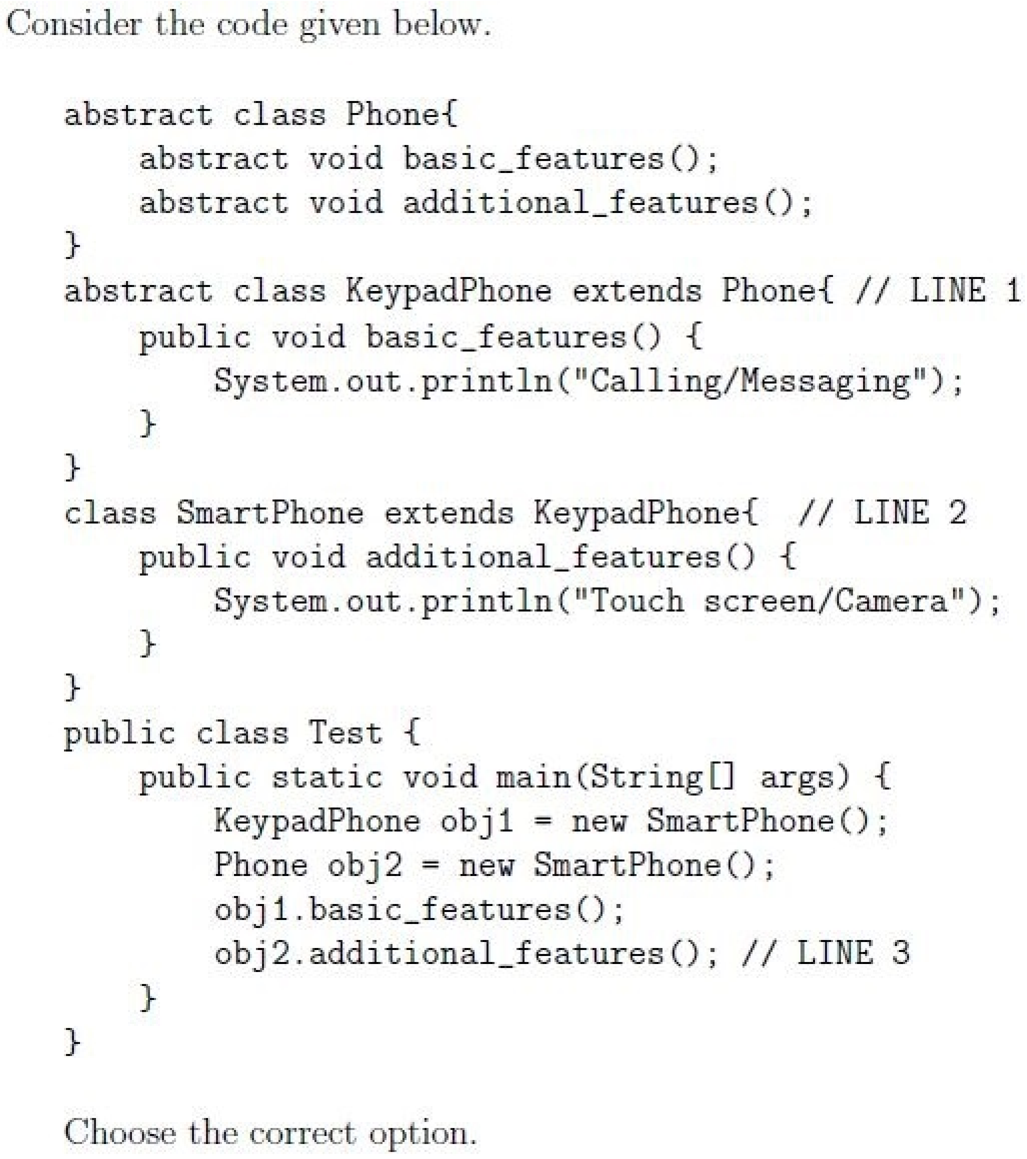
6406531929794. ✖ 

```
Default case
1 2 3
2 3
```

Question Number : 145 Question Id : 640653577897 Question Type : MCQ Is Question Mandatory : No Calculator : None Response Time : N.A Think Time : N.A Minimum Instruction Time : 0

Correct Marks : 3

Question Label : Multiple Choice Question

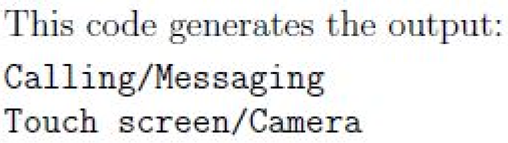


Consider the code given below.

```
abstract class Phone{
    abstract void basic_features();
    abstract void additional_features();
}
abstract class KeypadPhone extends Phone{ // LINE 1
    public void basic_features() {
        System.out.println("Calling/Messaging");
    }
}
class SmartPhone extends KeypadPhone{  // LINE 2
    public void additional_features() {
        System.out.println("Touch screen/Camera");
    }
}
public class Test {
    public static void main(String[] args) {
        KeypadPhone obj1 = new SmartPhone();
        Phone obj2 = new SmartPhone();
        obj1.basic_features();
        obj2.additional_features(); // LINE 3
    }
}
```

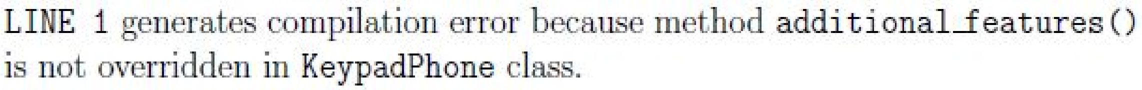
Choose the correct option.

Options :

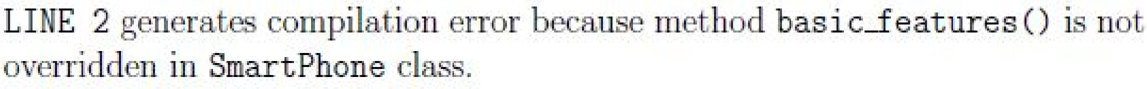
6406531929795. ✓ 

This code generates the output:

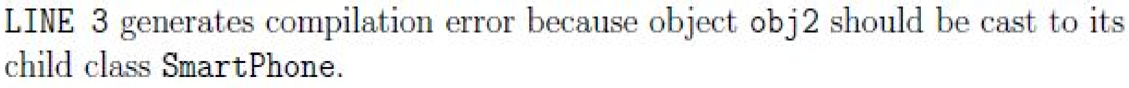
```
Calling/Messaging
Touch screen/Camera
```

6406531929796. ✗ 

LINE 1 generates compilation error because method `additional_features()` is not overridden in `KeypadPhone` class.

6406531929797. ✗ 

LINE 2 generates compilation error because method `basic_features()` is not overridden in `SmartPhone` class.

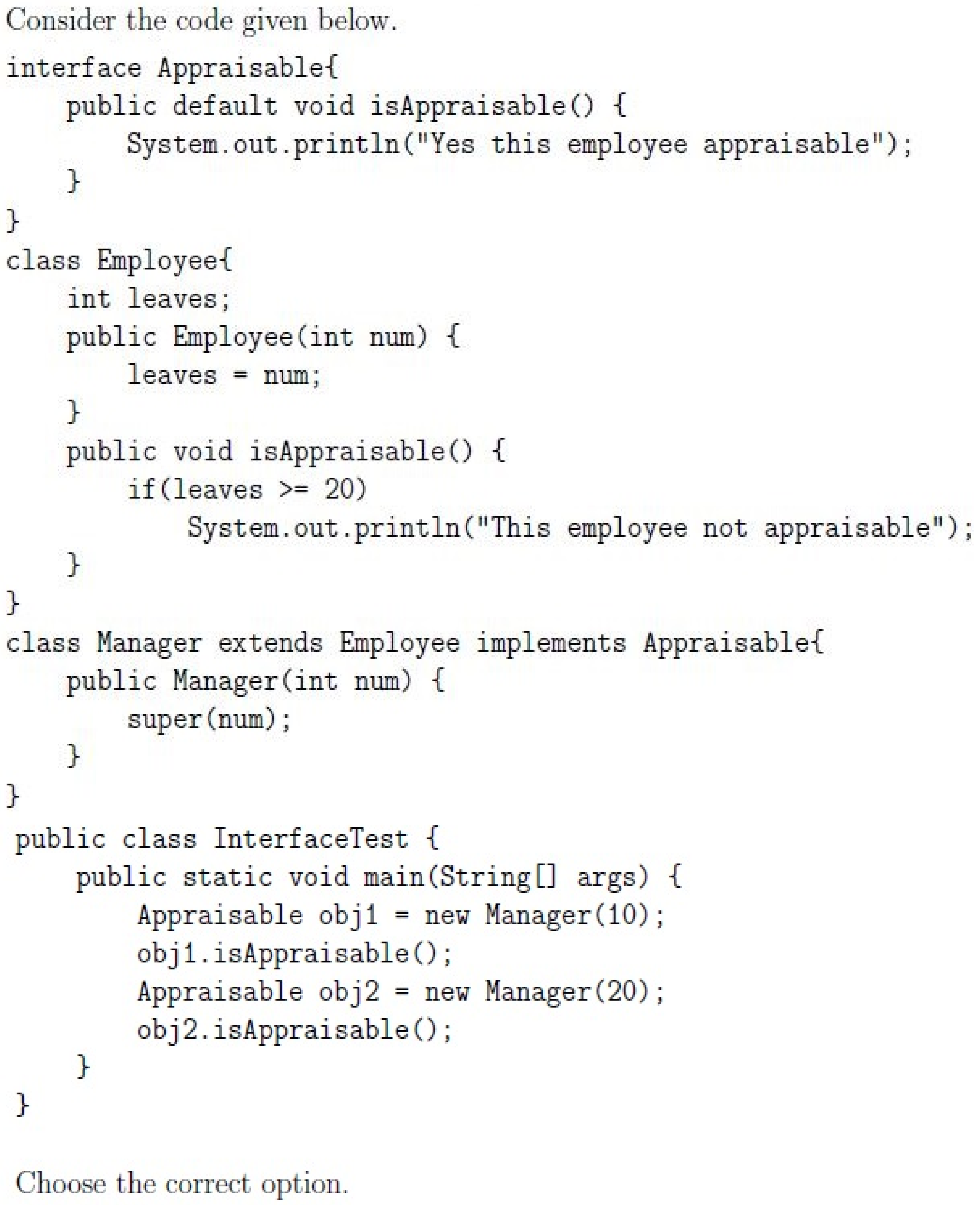
6406531929798. ✗ 

LINE 3 generates compilation error because object `obj2` should be cast to its child class `SmartPhone`.

Question Number : 146 Question Id : 640653577898 Question Type : MCQ Is Question Mandatory : No Calculator : None Response Time : N.A Think Time : N.A Minimum Instruction Time : 0

Correct Marks : 3

Question Label : Multiple Choice Question

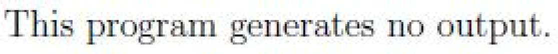


Consider the code given below.

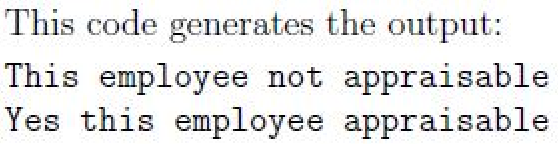
```
interface Appraisable{
    public default void isAppraisable() {
        System.out.println("Yes this employee appraisable");
    }
}
class Employee{
    int leaves;
    public Employee(int num) {
        leaves = num;
    }
    public void isAppraisable() {
        if(leaves >= 20)
            System.out.println("This employee not appraisable");
    }
}
class Manager extends Employee implements Appraisable{
    public Manager(int num) {
        super(num);
    }
}
 public class InterfaceTest {
     public static void main(String[] args) {
         Appraisable obj1 = new Manager(10);
         obj1.isAppraisable();
         Appraisable obj2 = new Manager(20);
         obj2.isAppraisable();
     }
 }
```

Choose the correct option.

Options :

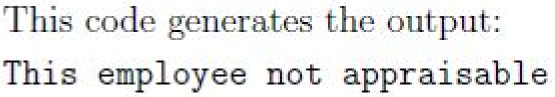
6406531929799. ✗ 

This program generates no output.

6406531929800. ✗ 

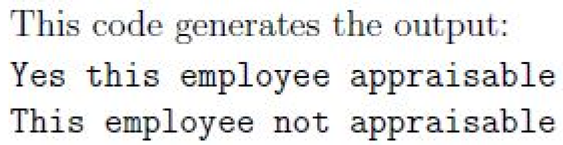
This code generates the output:

```
This employee not appraisable
Yes this employee appraisable
```

6406531929801. ✓ 

This code generates the output:

```
This employee not appraisable
```



This code generates the output:

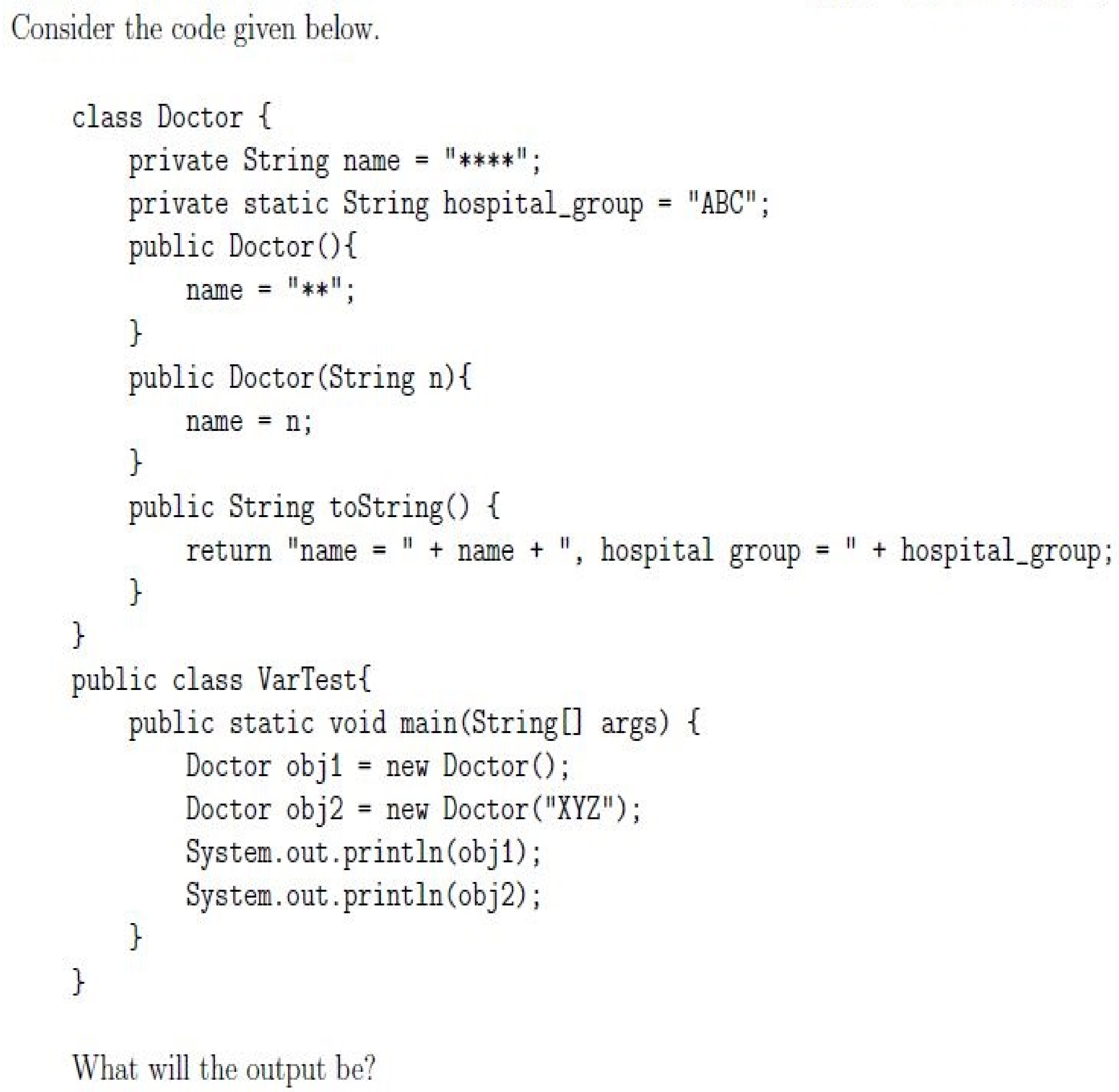
```
Yes this employee appraisable
This employee not appraisable
```

6406531929802. ✖

Question Number : 147 Question Id : 640653577899 Question Type : MCQ Is Question Mandatory : No Calculator : None Response Time : N.A Think Time : N.A Minimum Instruction Time : 0

Correct Marks : 3

Question Label : Multiple Choice Question

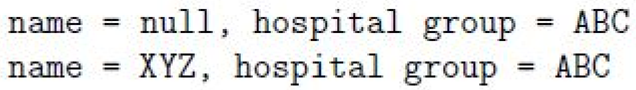


Consider the code given below.

```
class Doctor {
    private String name = "****";
    private static String hospital_group = "ABC";
    public Doctor(){
        name = "**";
    }
    public Doctor(String n){
        name = n;
    }
    public String toString() {
        return "name = " + name + ", hospital group = " + hospital_group;
    }
}
public class VarTest{
    public static void main(String[] args) {
        Doctor obj1 = new Doctor();
        Doctor obj2 = new Doctor("XYZ");
        System.out.println(obj1);
        System.out.println(obj2);
    }
}
```

What will the output be?

Options :



```
name = null, hospital group = ABC
name = XYZ, hospital group = ABC
```

6406531929803. ✖

6406531929804. ✗

```
name = **, hospital group = null
name = XYZ, hospital group = ABC
```

6406531929805. ✗

```
name = ****, hospital group = ABC
name = XYZ, hospital group = ABC
```

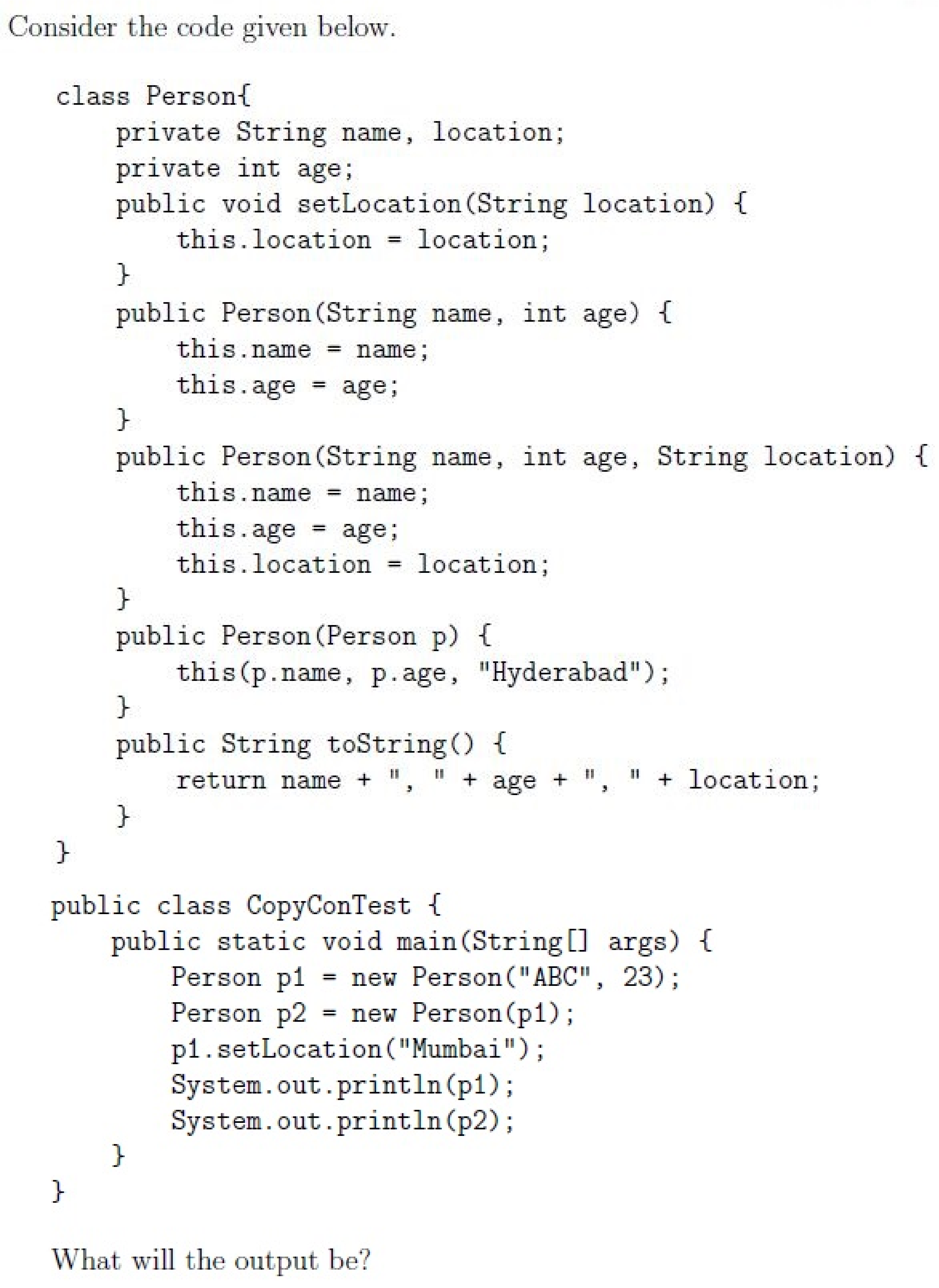
6406531929806. ✓

```
name = **, hospital group = ABC
name = XYZ, hospital group = ABC
```

Question Number : 148 Question Id : 640653577902 Question Type : MCQ Is Question Mandatory : No Calculator : None Response Time : N.A Think Time : N.A Minimum Instruction Time : 0

Correct Marks : 3

Question Label : Multiple Choice Question



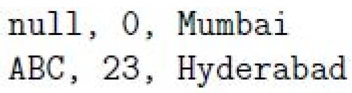
Consider the code given below.

```
class Person{
    private String name, location;
    private int age;
    public void setLocation(String location) {
        this.location = location;
    }
    public Person(String name, int age) {
        this.name = name;
        this.age = age;
    }
    public Person(String name, int age, String location) {
        this.name = name;
        this.age = age;
        this.location = location;
    }
    public Person(Person p) {
        this(p.name, p.age, "Hyderabad");
    }
    public String toString() {
        return name + ", " + age + ", " + location;
    }
}

public class CopyConTest {
    public static void main(String[] args) {
        Person p1 = new Person("ABC", 23);
        Person p2 = new Person(p1);
        p1.setLocation("Mumbai");
        System.out.println(p1);
        System.out.println(p2);
    }
}
```

What will the output be?

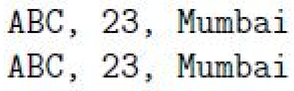
Options :

6406531929815. ✖



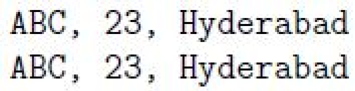
```
null, 0, Mumbai
ABC, 23, Hyderabad
```

6406531929816. ✖

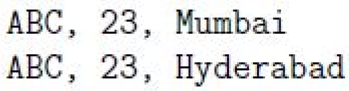


```
ABC, 23, Mumbai
ABC, 23, Mumbai
```

6406531929817. ✖



```
ABC, 23, Hyderabad
ABC, 23, Hyderabad
```

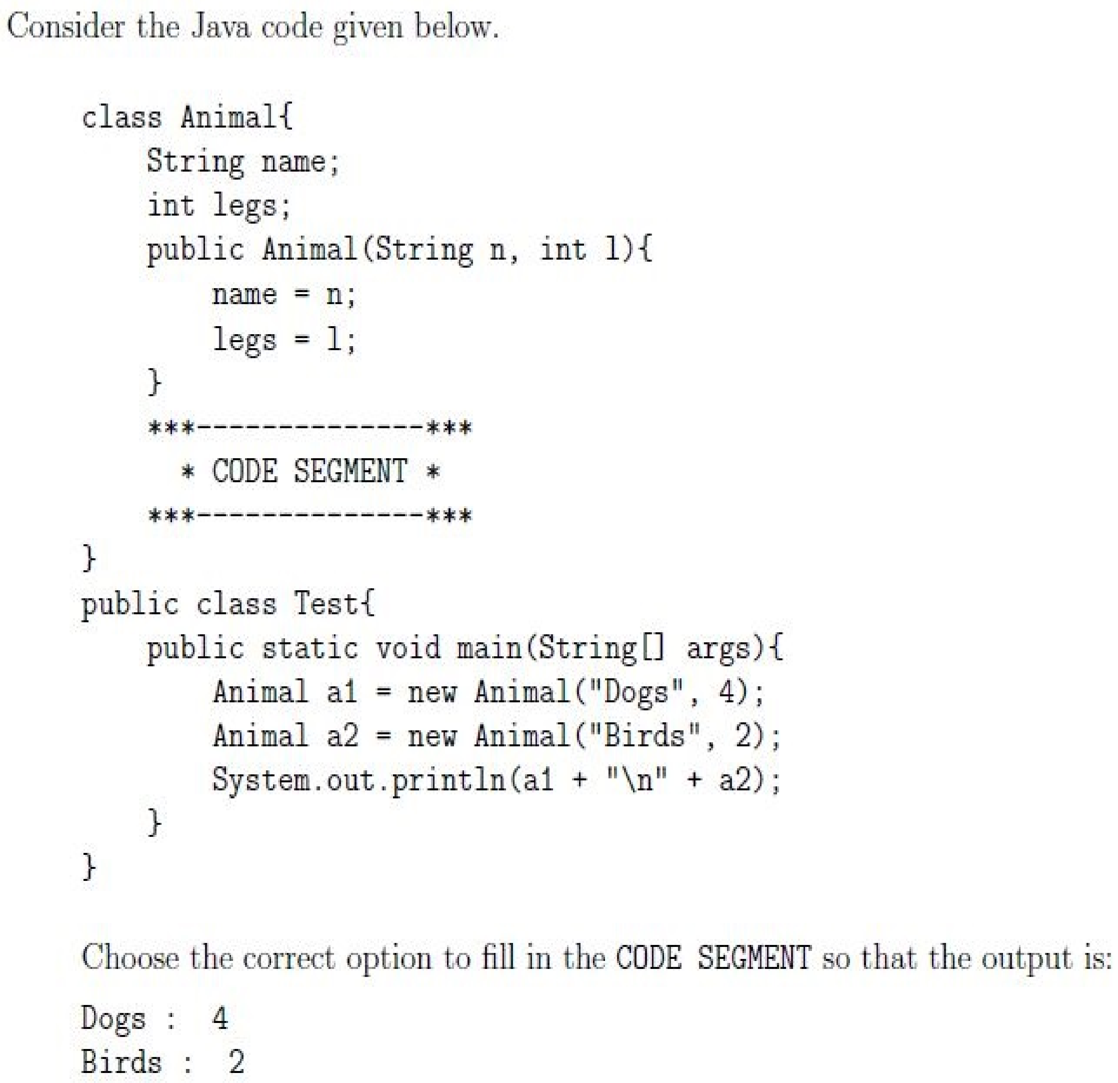
6406531929818. ✓ 

```
ABC, 23, Mumbai
ABC, 23, Hyderabad
```

Question Number : 149 Question Id : 640653577903 Question Type : MCQ Is Question Mandatory : No Calculator : None Response Time : N.A Think Time : N.A Minimum Instruction Time : 0

Correct Marks : 3

Question Label : Multiple Choice Question



Consider the Java code given below.

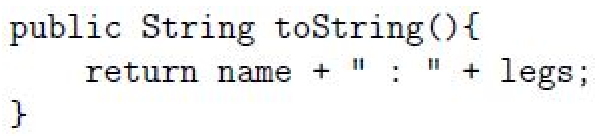
```
class Animal{
    String name;
    int legs;
    public Animal(String n, int l){
        name = n;
        legs = l;
    }
    ***---------------***
      * CODE SEGMENT *
    ***---------------***
}
public class Test{
    public static void main(String[] args){
        Animal a1 = new Animal("Dogs", 4);
        Animal a2 = new Animal("Birds", 2);
        System.out.println(a1 + "\n" + a2);
    }
}
```

Choose the correct option to fill in the CODE SEGMENT so that the output is:

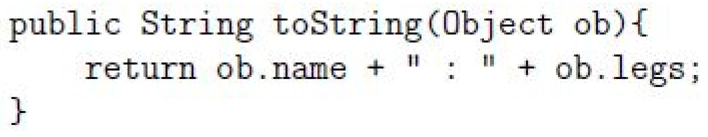
```
Dogs :  4
Birds :  2
```

Options :

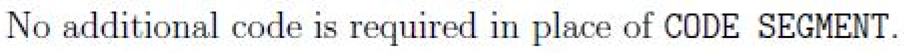
6406531929819. ✓



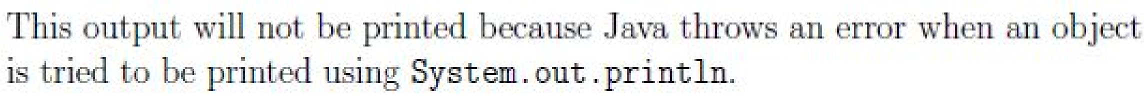
```
public String toString(){
    return name + " : " + legs;
}
```

6406531929820. ✖ 

```
public String toString(Object ob){
    return ob.name + " : " + ob.legs;
}
```

6406531929821. ✖ 

No additional code is required in place of `CODE SEGMENT`.

6406531929822. ✖ 

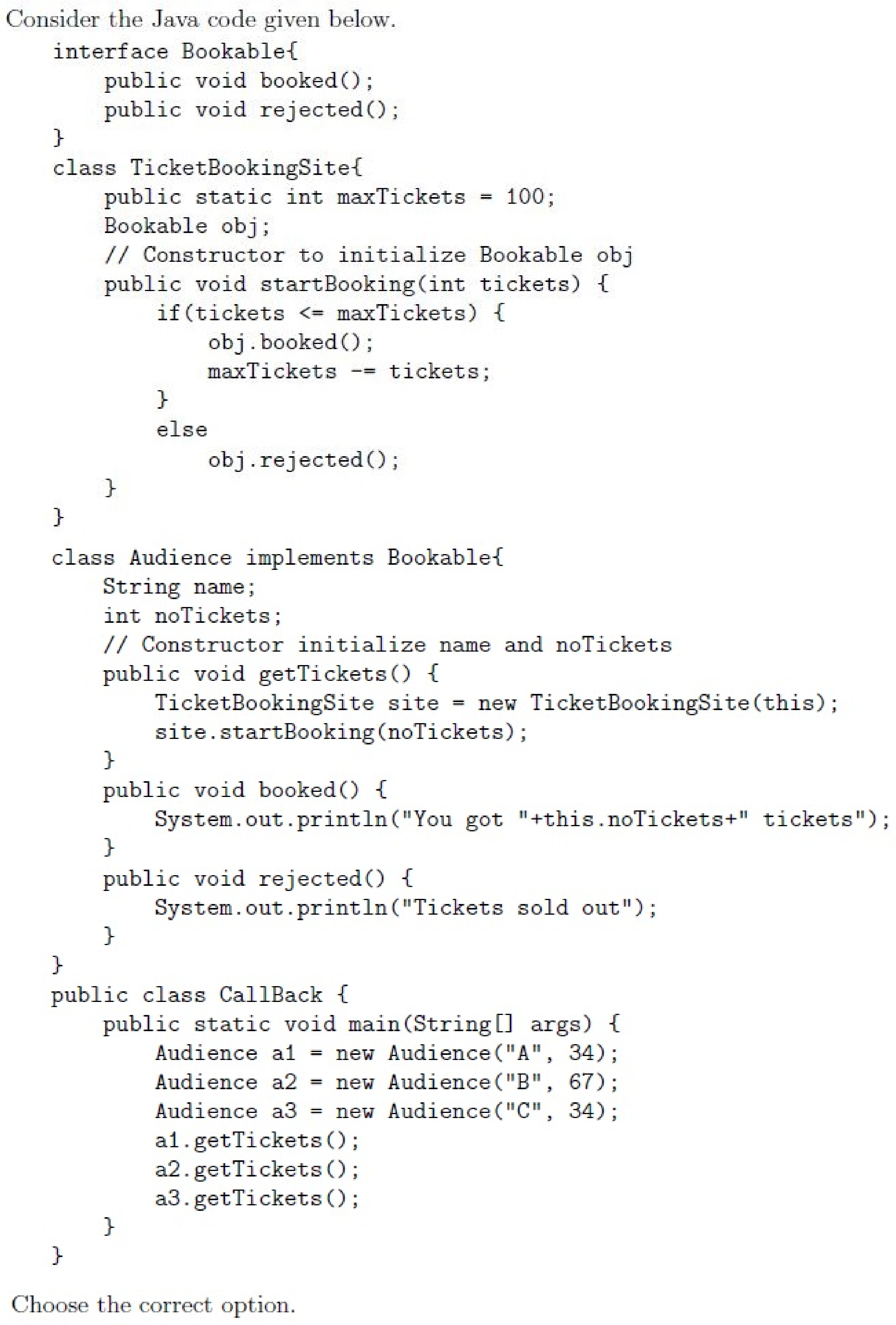
This output will not be printed because Java throws an error when an object is tried to be printed using `System.out.println`.

Sub-Section Number :	4
Sub-Section Id :	64065382605
Question Shuffling Allowed :	Yes
Is Section Default? :	null

Question Number : 150 Question Id : 640653577901 Question Type : MCQ Is Question Mandatory : No Calculator : None Response Time : N.A Think Time : N.A Minimum Instruction Time : 0

Correct Marks : 4

Question Label : Multiple Choice Question

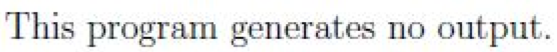


Consider the Java code given below.

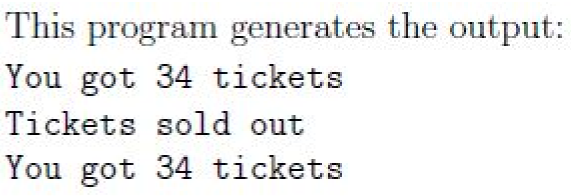
```
interface Bookable{
    public void booked();
    public void rejected();
}
class TicketBookingSite{
    public static int maxTickets = 100;
    Bookable obj;
    // Constructor to initialize Bookable obj
    public void startBooking(int tickets) {
        if(tickets <= maxTickets) {
            obj.booked();
            maxTickets -= tickets;
        }
        else
            obj.rejected();
    }
}

class Audience implements Bookable{
    String name;
    int noTickets;
    // Constructor initialize name and noTickets
    public void getTickets() {
        TicketBookingSite site = new TicketBookingSite(this);
        site.startBooking(noTickets);
    }
    public void booked() {
        System.out.println("You got "+this.noTickets+" tickets");
    }
    public void rejected() {
        System.out.println("Tickets sold out");
    }
}
public class CallBack {
    public static void main(String[] args) {
        Audience a1 = new Audience("A", 34);
        Audience a2 = new Audience("B", 67);
        Audience a3 = new Audience("C", 34);
        a1.getTickets();
        a2.getTickets();
        a3.getTickets();
    }
}
```

Choose the correct option.

Options :

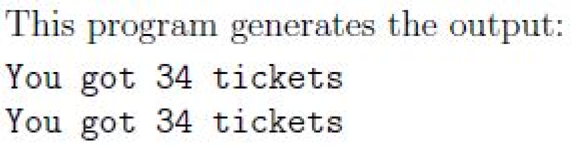
6406531929811. ✖ 

This program generates no output.

6406531929812. ✓ 

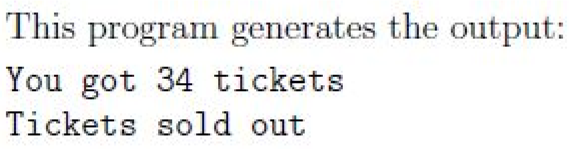
This program generates the output:

```
You got 34 tickets
Tickets sold out
You got 34 tickets
```

6406531929813. ✖ 

This program generates the output:

```
You got 34 tickets
You got 34 tickets
```

6406531929814. ✖ 

This program generates the output:

```
You got 34 tickets
Tickets sold out
```

AppDev2

Section Id :	64065339075
Section Number :	10
Section type :	Online
Mandatory or Optional :	Mandatory
Number of Questions :	17
Number of Questions to be attempted :	17
Section Marks :	50
Display Number Panel :	Yes
Group All Questions :	No
Enable Mark as Answered Mark for Review and Clear Response :	Yes
Maximum Instruction Time :	0
Sub-Section Number :	1
Sub-Section Id :	64065382606
Question Shuffling Allowed :	No