6406532774876. ✖ None of these are TRUE

Java

Section Id :	64065359213
Section Number :	6
Section type :	Online
Mandatory or Optional :	Mandatory
Number of Questions :	16
Number of Questions to be attempted :	16
Section Marks :	100
Display Number Panel :	Yes
Section Negative Marks :	0
Group All Questions :	No
Enable Mark as Answered Mark for Review and Clear Response :	No
Section Maximum Duration :	0
Section Minimum Duration :	0
Section Time In :	Minutes
Maximum Instruction Time :	0
Sub-Section Number :	1
Sub-Section Id :	640653122738
Question Shuffling Allowed :	No

Question Number : 83 Question Id : 640653825076 Question Type : MCQ

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 :

6406532774877. ✔ YES

6406532774878. ✖ NO

Sub-Section Number :	2
Sub-Section Id :	640653122739
Question Shuffling Allowed :	Yes

Question Number : 84 Question Id : 640653825077 Question Type : MSQ

Correct Marks : 6 Max. Selectable Options : 0

Question Label : Multiple Select Question

Which of the following statements is/are correct about activation records?

Options :

6406532774879. ✖ Return value link points to the start of the previous activation record.

6406532774880. ✔ An activation record is pushed into the stack when a function is called, and is popped out when the function returns.

6406532774881. ✖ The variables present in every activation record in the stack are in scope and are accessible.

6406532774882. ✔ The variables present in the topmost activation record of the stack are in scope and are accessible.

Sub-Section Number : 3

Sub-Section Id : 640653122740

Question Shuffling Allowed : Yes

Question Number : 85 Question Id : 640653825078 Question Type : MCQ

Correct Marks : 6

Question Label : Multiple Choice Question

Match the following terms with their descriptions/properties.

Terms	Properties
1. State	A. Determine the choice of method implementation at run time
2. Behaviour	B. Compatibility of interfaces
3. Subtyping	C Determined by the information in the instance variables
4. Inheritance	D. Methods that operate on an object
5. Dynamic lookup	E. Reuse of implementations

Options :

6406532774883. ✖ 1-B, 2-A, 3-C, 4-D, 5-E

6406532774884. ✖ 1-A, 2-B, 3-D, 4-E, 5-C

6406532774885. ✔ 1-C, 2-D, 3-B, 4-E, 5-A

6406532774886. ✖ 1-D, 2-C, 3-E, 4-A, 5-B

Question Number : 86 Question Id : 640653825079 Question Type : MCQ

Correct Marks : 6

Question Label : Multiple Choice Question

Consider the Java code given below.

```
public class Test{
    public static void main(String[] args){
        int a[] = {10, 20};
        int x = a[0];
        for(int i : a){
            switch(i){
                case 10:
                    x = x + 10;
                    System.out.println(x);
                    break;
                case 20:
                    x = x + 20;
                    System.out.println(x);
                    break;
                default:
                    System.out.println(x);
            }
        }
    }
}
```

What will the output be?

Options :

6406532774887. ✖ 20
30
30

6406532774888. ✔ 20
40

6406532774889. ✖ 20
40
60

6406532774890. ✖ 10
20
40

Question Number : 87 Question Id : 640653825080 Question Type : MCQ
Correct Marks : 6
Question Label : Multiple Choice Question

Consider the Java code given below.

```
interface Printer{
    public default void print(){
        System.out.println("Prints");
    }
}
interface Scanner{
    public default void scan(){
        System.out.println("Scans");
    }
}
class Device implements Printer, Scanner{
    public void print(){
        System.out.println("Color printing");
    }
}
public class Test {
    public static void main(String[] args) {
        Printer p1 = new Device();
        p1.print();
        p1.scan();  //LINE 1
    }
}
```

Choose the correct option.

Options :

6406532774891. ✖ This program generates the output:
`Color printing`
`Scans`

6406532774892. ✖ This program generates the output:
`Prints`
`Scans`

6406532774893. ✔ `LINE 1` generates compiler error because `p1` of type `Printer` cannot invoke method `scan()`.

6406532774894. ✖ This program generates compiler error because neither is class `Device` declared as abstract nor does it override method `scan()`.

Question Number : 88 Question Id : 640653825084 Question Type : MCQ
Correct Marks : 6

Question Label : Multiple Choice Question

Consider the Java code given below.

```
class Employee {
    public void performTasks() {
        System.out.println("Perform tasks");
    }
}
class Manager extends Employee {
    public void plan() {
        System.out.println("Plan tasks");
    }

    public void monitor() {
        System.out.println("Monitor employees");
    }
}
class DeliveryHead extends Manager {
    public void plan() {
        System.out.println("Plan delivery operations");
    }
}
public class Test {
    public static void main(String[] args) {
        Manager obj = new DeliveryHead();
        obj.performTasks(); // LINE 1
        obj.monitor();
        obj.plan(); // LINE 2
    }
}
```

Choose the correct option.

Options :

6406532774907. ✖ LINE 1 generates compilation error because method `performTasks()` cannot be invoked on `obj`.

6406532774908. ✖ This code generates the below output followed by runtime Error at LINE 2 because there is ambiguity in which `plan( )` method is being invoked.

```
Perform tasks
Monitor employees
```

6406532774909. ✔ This code generates the output:

```
Perform tasks
Monitor employees
Plan delivery operations
```

This code generates the output:

```
Perform tasks
Monitor employees
Plan tasks
```

6406532774910. ✖

Question Number : 89 Question Id : 640653825090 Question Type : MCQ
Correct Marks : 6

Question Label : Multiple Choice Question

Consider the Java code given below.

```
class Bird {
    private String species;
    private String color;
    public Bird(String species, String color) {
        this.species = species;
        this.color = color;
    }
    // ------- CODE SEGMENT ---------
}
public class Test {
    public static void main(String[] args) {
        Bird b1 = new Bird("Parrot", "Green");
        Bird b2 = new Bird("Sparrow", "Brown");
        System.out.println(b1 + "\n" + b2);
    }
}
```

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

```
Parrot :  Green
Sparrow :  Brown
```

Options :

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

6406532774931. ✖

```
public String toString() {
    return species + " : " + color;
}
```

6406532774932. ✔

No additional code is required in place of CODE SEGMENT.

6406532774933. ✖

6406532774934. ✖ 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 : 640653122741
Question Shuffling Allowed : Yes

Question Number : 90 Question Id : 640653825081 Question Type : MCQ
Correct Marks : 7
Question Label : Multiple Choice Question

Consider the Java code given below. Identify the correct statement to fill in the blank at LINE 1, such that the output is: Eligible for Diploma

```
interface Eligibility {
    void printEligibility();
}
class Student {
    private double cgpa;
    //Constructor to initialize instance variable
    public Eligibility checkEligibility() {
        if (cgpa > 6.0)
            return new Eligible();
        return new NotEligible();
    }
    private class Eligible implements Eligibility {
        public void printEligibility() {
            System.out.println("Eligible for Diploma");
        }
    }
    private class NotEligible implements Eligibility {
        public void printEligibility() {
            System.out.println("Not eligible for Diploma");
        }
    }
}
public class Test {
    public static void main(String[] args) {
        Student s1 = new Student(5.5);
        Student s2 = new Student(7.5);
        ----------------------------    //LINE 1
    }
}
```

Options :

6406532774895. ✔

```
s2.checkEligibility().printEligibility();
```

6406532774896. ✖
```
s2.printEligibility();
```

6406532774897. ✖
```
s1.printEligibility();
```

6406532774898. ✖
```
s1.checkEligibility().printEligibility();
```

Question Number : 91 Question Id : 640653825082 Question Type : MCQ
Correct Marks : 7
Question Label : Multiple Choice Question

Consider the code given below that checks whether two phones are the same. Method `equals` is overridden to compare two `Phone` objects as follows.
If two phones have the same `brand` and `imeiNumber`, then they are the same.

```
class Phone {
    private String brand;
    private int imeiNumber;

    // Constructor to initialize instance variables

    public boolean equals(Object obj) {
        //CODE BLOCK
    }
}
public class Test {
    public static void main(String[] args) {
        Phone p1 = new Phone("Samsung", 123456);
        Phone p2 = new Phone("Samsung", 123456);
         if (p1.equals(p2))
            System.out.println("p1 and p2 are same");
        else
            System.out.println("p1 and p2 are different");
    }
}
```

Choose the correct option(s) to fill in place of `CODE BLOCK` so that the output is:
`p1 and p2 are same`

Options :

6406532774899. ✖

```
if(obj instanceof Phone) {
    if(this.brand == obj.brand && this.imeiNumber == obj.imeiNumber)
        return true;
}
return false;
```

6406532774900. ✖

```
if(this.brand == obj.brand && this.imeiNumber == obj.imeiNumber)
    return true;
return false;
```

6406532774901. ✖

```
if(obj instanceof Phone) {
    Phone p = obj;
    if(this.brand == p.brand && this.imeiNumber == p.imeiNumber)
        return true;
}
return false;
```

6406532774902. ✔

```
if(obj instanceof Phone) {
    Phone p = (Phone) obj;
    if(this.brand == p.brand && this.imeiNumber == p.imeiNumber)
        return true;
}
return false;
```

Question Number : 92 Question Id : 640653825083 Question Type : MCQ
Correct Marks : 7
Question Label : Multiple Choice Question

Consider the Java code given below.

```
class TravelAgency{
  String name;
  String[] destinations;

  public TravelAgency(String n, String[] d){
    name = n;
    destinations = d;
  }
  public TravelAgency(TravelAgency t){
    this.name = t.name;
    this.destinations = t.destinations;
  }
}
public class Test{
  public static void main(String[] args){
    String[] d  = {"Ooty", "Bali", "Thailand"};
    TravelAgency t1 = new TravelAgency("RoamWorld", d);
    TravelAgency t2 = new TravelAgency(t1);
    t2.name= "ValleyTravel";
    t2.destinations[0] = "Goa";
    System.out.println(t1.name + "," +t1.destinations[0]);
    System.out.println(t2.name + "," +t2.destinations[0]);
  }
}
```

What will the output be?

Options :

6406532774903. ✖
```
ValleyTravel,Goa
ValleyTravel,Goa
```

6406532774904. ✔
```
RoamWorld,Goa
ValleyTravel,Goa
```

6406532774905. ✖
```
RoamWorld,Ooty
ValleyTravel,Goa
```

6406532774906. ✖
```
RoamWorld,Ooty
RoamWorld,Goa
```

Question Number : 93 Question Id : 640653825085 Question Type : MCQ
Correct Marks : 7

Question Label : Multiple Choice Question

Consider the Java code given below.

```
class Screen {
    void showContent() {
        System.out.println("Display on screen");
    }
}
class TV extends Screen {
    void showContent() {
        super.showContent();
        System.out.println("Display on TV");
    }
    void showContent(String s) {
        System.out.println("Display on TV: " + s);
    }
}
class Monitor extends TV {
    void showContent() {
        super.showContent();
        System.out.println("Display on monitor");
    }
    void showContent(String s) {
        System.out.println("Display on monitor: " + s);
    }
}
class Test {
    public static void main(String[] args) {
        TV obj = new Monitor();  // LINE 1
        obj.showContent();
        obj.showContent("News");  // LINE 2
    }
}
```

Options :

6406532774911. ✔ The program generates output:

```
Display on screen
Display on TV
Display on monitor
Display on monitor: News
```

6406532774912. ✖ LINE 1 generates compilation error because a variable of type TV cannot refer to an object of type Monitor.

6406532774913. ✖

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

```
Display on TV
Display on monitor
Display on monitor: News
```

The program generates output:

```
Display on screen
Display on TV
Display on TV: News
```

6406532774914. ✖

Question Number : 94 Question Id : 640653825086 Question Type : MCQ
Correct Marks : 7

Question Label : Multiple Choice Question

Consider the Java code given below.

```
1  interface Coolable {
2      public void startCooling();
3      public void stopCooling();
4  }
5  interface TemperatureAdjustable {
6      public void adjustTemperature();
7      default void displayTemperature() {
8          System.out.println("Temperature is adjustable.");
9      }
10 }
11 class AirConditioner implements Coolable, TemperatureAdjustable {
12     public void startCooling() {
13         System.out.println("Cooling started.");
14     }
15     public void stopCooling() {
16         System.out.println("Cooling stopped.");
17     }
18 }
```

Choose the correct option regarding the above code.

Options :

LINE 7 generates compilation error because the method displayTemperature is not abstract.

6406532774915. ✖

LINE 11 generates compilation error because class AirConditioner cannot implement two interfaces.

6406532774916. ✖

6406532774917. ✔ LINE 11 generates compilation error because class `AirConditioner` is not declared as abstract.

6406532774918. ✖ LINE 2, LINE 3 & LINE 6 generate compilation errors because the methods `startCooling()`, `stopCooling()` and `adjustTemperature()` are not abstract.

Question Number : 95 Question Id : 640653825087 Question Type : MCQ
Correct Marks : 7
Question Label : Multiple Choice Question

Consider the Java code given below.

```
interface Singable {
    default void sing() {
        System.out.println("Sings");
    }
    public void perform();
}
abstract class Musician implements Singable {
    public void sing() {
        System.out.println("Sings song");
    }
}
class LeadSinger extends Musician {
    public void perform() {
        System.out.println("Leads group");
    }
}
public class Test {
    public static void main(String[] args) {
        Singable obj1 = new LeadSinger(); // LINE 1
        Musician obj2 = new LeadSinger();
        obj2.sing();
        obj2.perform(); // LINE 2
    }
}
```

Choose the correct option.

Options :

6406532774919. ✔ This code generates the output:

```
Sings song
Leads group
```

6406532774920. ✖ This code generates the output:

```
Sings
Leads group
```

6406532774921. ✖ LINE 1 generates compilation error because a variable of type `Singable` cannot refer to an object of type `LeadSinger`.

6406532774922. ✖ LINE 2 generates compilation error because the method `perform()` cannot be invoked on `obj2`

Question Number : 96 Question Id : 640653825089 Question Type : MCQ
Correct Marks : 7
Question Label : Multiple Choice Question

Consider the Java code given below.

```
class Player {
    private String name;
    private int jerseyNumber;
    public Player(String nm){
        name = nm;
    }
    public Player(int number){
        jerseyNumber = number;
    }
    public Player(String nm, int number) {
        name = nm;
        jerseyNumber = number;
    }
    public String toString() {
        return "Name: " + name + ", Jersey Number: " + jerseyNumber;
    }
}

class Captain extends Player {
    private String team;
    // ------- CODE BLOCK ---------
    public String toString() {
        return super.toString() + ", Team: " + team;
    }
}
```

Choose the correct option to fill in place of `CODE BLOCK` to instantiate instance variables of class `Captain`

Options :

6406532774927. ✖
```
public Captain(String t) {
    team = t;
}
```

6406532774928. ✔
```
public Captain(String nm, int number, String t) {
        super(nm, number);
        team = t;
}
```

6406532774929. ✖
```
public Captain(String nm, int number, String t) {
    name = nm;
    jerseyNumber = number;
    team = t;
}
```

6406532774930. ✖
```
public Captain(String nm, int number, String t) {
        team = t;
        super(nm, number);
}
```

Question Number : 97 Question Id : 640653825091 Question Type : MCQ
Correct Marks : 7
Question Label : Multiple Choice Question

Consider the Java code given below.

```
interface Iterator {
    public boolean has_next();
    public Object get_next();
}
abstract class Printable {
    public abstract void print();
}
class BankAccount extends Printable {
    private String accountName;
    private double balance;
    public BankAccount(String aN, double b) {
        //initialize accountname and balance
    }
    public void print() {
        System.out.println(accountName + ", " + balance);
    }
}
class BankAccountList {
    private final int limit = 3;
    private BankAccount[] list;
    public BankAccountList(BankAccount[] accounts) {
        this.list = accounts;
    }
    private class BankAccountIter implements Iterator {
        private int indx;
        public BankAccountIter() {
            //constructor
        }
        public boolean has_next() {
            //if next element available in list return true;else false
        }
        public Object get_next() {
            //return next element from list
        }
    }
    public Iterator getIterator() {
        return new BankAccountIter();
    }
}
public class Test{
    public static void main(String[] args) {
        BankAccountList.BankAccount[] accounts = {
            new BankAccountList.BankAccount("Priya", 500),
            new BankAccountList.BankAccount("Ravi", 1000),
            new BankAccountList.BankAccount("Suresh", 1500)
        };
        BankAccountList bList = new BankAccountList(accounts);
        Iterator iter = bList.getIterator();
        while(iter.has_next()) {
           ------------------;      //LINE 1
        }
    }
}
```

Identify the appropriate statement to fill in the blank at LINE 1, such that the output is:

```
Priya, 500
Ravi, 1000
Suresh, 1500
```

Options :

6406532774935. ✔ `((Printable)iter.get_next()).print()`

6406532774936. ✖ `((BankAccount)iter.get_next()).print()`

6406532774937. ✖ `((BankAccountList)iter.get_next()).print()`

6406532774938. ✖ `iter.get_next().print();`

Sub-Section Number : 5
Sub-Section Id : 640653122742
Question Shuffling Allowed : Yes

Question Number : 98 Question Id : 640653825088 Question Type : MCQ
Correct Marks : 8
Question Label : Multiple Choice Question

Consider the Java code given below.

```
class Appliance {
    private int code;
    private static double basePower = 100;
    public Appliance(int c) {
        code = c;
    }
    public final double electricityFare() {
        return basePower*2;
    }
}
class Cooler extends Appliance {
    public Cooler(int x) {
        super(x);
    }
    public final double electricityFare() { //LINE 1
        return basePower*3;                  //LINE 2
    }
}
public class Test {
    public static void main(String[] args) {
        Appliance d1 = new Cooler(101); // LINE 3
        Cooler c1 = new Appliance(105); // LINE 4
        d1.electricityFare();
        c1.electricityFare();
    }
}
```

Which of the following statements is FALSE?

Options :

6406532774923. ✖ LINE 1 generates compilation error because the method `electricityFare()` cannot be overridden.

6406532774924. ✖

LINE 2 generates compilation error because instance variable basePower cannot be accessed in class Cooler.

6406532774925. ✔ LINE 3 generates compilation error because a variable of type Appliance cannot refer to an object of type Cooler.

6406532774926. ✖ LINE 4 generates compilation error because a variable of type Cooler cannot refer to an object of type Appliance.

AppDev2

Section Id :	64065359214
Section Number :	7
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
Section Negative Marks :	0
Group All Questions :	No
Enable Mark as Answered Mark for Review and Clear Response :	No
Section Maximum Duration :	0
Section Minimum Duration :	0
Section Time In :	Minutes
Maximum Instruction Time :	0
Sub-Section Number :	1
Sub-Section Id :	640653122743
Question Shuffling Allowed :	No

Question Number : 99 Question Id : 640653825092 Question Type : MCQ

Correct Marks : 0

Question Label : Multiple Choice Question

THIS IS QUESTION PAPER FOR THE SUBJECT "DIPLOMA LEVEL : MODERN APPLICATION DEVELOPMENT II (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.