

Response Type : Numeric

Evaluation Required For SA : Yes

Show Word Count : Yes

Answers Type : Range

Text Areas : PlainText

Possible Answers :

3.70 to 3.78

Java

Section Id :	64065344903
Section Number :	9
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 :	Yes
Maximum Instruction Time :	0
Sub-Section Number :	1
Sub-Section Id :	64065395178
Question Shuffling Allowed :	No
Is Section Default? :	null

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 :

6406532239916. ✓ YES

6406532239917. ✗ NO

Sub-Section Number :

2

Sub-Section Id :

64065395179

Question Shuffling Allowed :

Yes

Is Section Default? :

null

Question Number : 135 Question Id : 640653668576 Question Type : MSQ Is Question

Mandatory : No Calculator : None Response Time : N.A Think Time : N.A Minimum Instruction

Time : 0

Correct Marks : 6 Max. Selectable Options : 0

Question Label : Multiple Select Question

Which of the following is/are CORRECT about heap?

Choose the correct option(s).

Options :

6406532239918. ✓ Heap is used to store dynamically allocated data.

6406532239919. ✗ Activation records created for functions are allocated in heap.

6406532239920.

✔ Heap storage needs to be explicitly requested by the programmer.

6406532239921. ✖ A heap storage allocation cannot be deallocated in any way.

Sub-Section Number : 3

Sub-Section Id : 64065395180

Question Shuffling Allowed : Yes

Is Section Default? : null

Question Number : 136 Question Id : 640653668577 Question Type : MCQ Is Question

Mandatory : No Calculator : None Response Time : N.A Think Time : N.A Minimum Instruction Time : 0

Correct Marks : 6

Question Label : Multiple Choice Question

Match the following terms with their descriptions/properties.

Terms	Properties
1. Abstraction	A. Determine the choice of method implementation at run time
2. Subtyping	B. Increases the readability and reusability
3. Modularity	C. Public interface, private implementation
4. Inheritance	D. Compatibility of interfaces
5. Dynamic lookup	E. Reuse of implementations

Options :

6406532239922. ✖ 1-B, 2-A, 3-C, 4-D, 5-E

6406532239923. ✖ 1-A, 2-B, 3-D, 4-E, 5-C

6406532239924. ✔ 1-C, 2-D, 3-B, 4-E, 5-A

6406532239925. ✖ 1-D, 2-C, 3-E, 4-A, 5-B

Question Number : 137 Question Id : 640653668578 Question Type : MCQ Is Question

Mandatory : No Calculator : None Response Time : N.A Think Time : N.A Minimum Instruction Time : 0

Correct Marks : 6

Question Label : Multiple Choice Question

Consider the Java program below.

```
public class FClass{
    public static void main(String[] args){
        int a[] = {10, 20, 30};
        int x = 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;
                case 30:
                    x = x + 30;
                    System.out.println(x);
                    break;
                default:
                    System.out.println(x);
            }
        }
    }
}
```

What will the output be?

Options :

10

40

6406532239926. ✖ 70

10

30

6406532239927. ✔ 60

6406532239928. ✖

10
30
60
60

10
30
60
10

6406532239929. ✖

Question Number : 138 Question Id : 640653668579 Question Type : MCQ Is Question Mandatory : No Calculator : None Response Time : N.A Think Time : N.A Minimum Instruction Time : 0

Correct Marks : 6

Question Label : Multiple Choice Question

Consider the Java code given below.

```
interface Compiler{
    public default void compile(){
        System.out.println("Compiles");
    }
}
interface Interpreter{
    public default void interpret(){
        System.out.println("Interprets");
    }
}
class Language implements Compiler, Interpreter{
    public void compile(){
        System.out.println("Language specific compiler");
    }
}
public class Test {
    public static void main(String[] args) {
        Compiler c1 = new Language();
        c1.compile();
        c1.interpret();
    }
}
```

Choose the correct option.

Options :

This program generates the output:

Language specific compiler

6406532239930. ✖ Interprets

This program generates the output:

Compiles

6406532239931. ✖ Interprets

This program generates compiler error because c1 of type Compiler cannot invoke method interpret().

6406532239932. ✔

This program generates compiler error because neither is class Language declared as abstract nor does it override method interpret().

6406532239933. ✖

Question Number : 139 Question Id : 640653668583 Question Type : MCQ Is Question

Mandatory : No Calculator : None Response Time : N.A Think Time : N.A Minimum Instruction Time : 0

Correct Marks : 6

Question Label : Multiple Choice Question

Consider the code given below.

```

class Student{
    public void audit() {
        System.out.println("Audit classes");
    }
}
class TeamLeader extends Student{
    public void monitor() {
        System.out.println("Monitors team");
    }
    public void mentor() {
        System.out.println("Mentors team");
    }
}
class SchoolLeader extends TeamLeader{
    public void monitor() {
        System.out.println("Monitors school");
    }
}
public class Test{
    public static void main(String[] args) {
        TeamLeader obj = new SchoolLeader();
        obj.audit(); //LINE 1
        obj.mentor();
        obj.monitor(); //LINE 2
    }
}

```

Choose the correct option.

Options :

6406532239946. ✖ LINE 1 generates compilation error because method audit() cannot be invoked on obj.

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

Audit classes

6406532239947. ✖ Mentors team

This code generates the output:

Audit classes

Mentors team

6406532239948. ✔ Monitors school

This code generates the output:

Audit classes

Mentors team

6406532239949. ✖ Monitors team

Question Number : 140 Question Id : 640653668585 Question Type : MCQ Is Question

Mandatory : No Calculator : None Response Time : N.A Think Time : N.A Minimum Instruction Time : 0

Correct Marks : 6

Question Label : Multiple Choice Question

Consider the Java code given below.

```
interface Displayable{
    public default void display(){
        System.out.println("Displays digits");
    }
}
class Calculator implements Displayable{           //LINE 1
}
class SmartCalculator implements Displayable{
    public void display() {
        System.out.println("Displays alphabets and digits");
    }
}
public class Test{
    public static void main(String[] args) {
        Displayable d1 = new Calculator();
        d1.display();                               // LINE 2
        Displayable d2 = new SmartCalculator();
        d2.display();
    }
}
```

Choose the correct option.

Options :

6406532239954. ✖ Compiler error at LINE 1 because class Calculator is not abstract

This program generates output:
Displays digits
6406532239955. ✓ Displays alphabets and digits

This program generates output:
Displays digits
6406532239956. ✗ Displays digits

LINE 2 generates runtime error because method (display) is not defined for
6406532239957. ✗ class Calculator

Question Number : 141 Question Id : 640653668589 Question Type : MCQ Is Question Mandatory : No Calculator : None Response Time : N.A Think Time : N.A Minimum Instruction Time : 0

Correct Marks : 6

Question Label : Multiple Choice Question

Consider the code given below.

```

class Student{
    private String name;
    private int rollNo;
    public Student(String n, int r){
        name = n;
        rollNo = r;
    }
    /**-----***/
    /* CODE SEGMENT */
    /**-----***/
}
public class Test{
    public static void main(String[] args){
        Student s1 = new Student("Radha", 1);
        Student s2 = new Student("Raju", 2);
        System.out.println(s1 + "\n" + s2);
    }
}

```

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

Radha : 1

Raju : 2

Options :

```

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

```

6406532239970. ✖

```

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

```

6406532239971. ✔

6406532239972. ✖ No additional code is required in place of CODE SEGMENT.

This output will not be printed because Java throws an error when an object

6406532239973. ✖ is tried to be printed using System.out.println.

Sub-Section Number :

4

Sub-Section Id :

64065395181

Question Shuffling Allowed :

Yes

Is Section Default? :

null

Question Number : 142 Question Id : 640653668580 Question Type : MCQ Is Question

Mandatory : No Calculator : None Response Time : N.A Think Time : N.A Minimum Instruction Time : 0

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: Discount available

```
interface Enquiry{
    public void printAvailability();
}
class Customer{
    // ...
    public Enquiry checkAmount(){
        // ...
        if(bill > 5000)
            return new PassedEnquiry();
        return new FailedEnquiry();
    }
    private class PassedEnquiry implements Enquiry{
        public void printAvailability(){
            System.out.println("Discount available");
        }
    }
    private class FailedEnquiry implements Enquiry{
        public void printAvailability(){
            System.out.println("No discount");
        }
    }
}

public class Test {
    public static void main(String[] args) {
        Customer c1 = new Customer(4500);           //bill of c1 is 4500
        Customer c2 = new Customer(7000);           //bill of c2 is 7000
        -----                                     //LINE 1
    }
}
```

Options :

6406532239934. ✓ c2.checkAmount().printAvailability();

6406532239935. ✖ c2.printAvailability();

6406532239936. ✖ c1.printAvailability();

6406532239937. ✖ c1.checkAmount().printAvailability();

Question Number : 143 Question Id : 640653668582 Question Type : MCQ Is Question

Mandatory : No Calculator : None Response Time : N.A Think Time : N.A Minimum Instruction Time : 0

Correct Marks : 7

Question Label : Multiple Choice Question

Consider the Java code given below.

```
class Student{
    String name;
    int[] marks;

    public Student(String n, int[] m){
        name = n;
        marks = m;
    }
    public Student(Student s){
        this.name = s.name;
        this.marks = s.marks;
    }
}

public class Test{
    public static void main(String[] args){
        int[] m = {50, 70, 60};
        Student s1 = new Student("Siva", m);
        Student s2 = new Student(s1);
        s2.name= "Krishna";
        s2.marks[0] = 80;
        System.out.println(s1.name + "," +s1.marks[0]);
        System.out.println(s2.name + "," +s2.marks[0]);
    }
}
```

What will the output be?

Options :

6406532239942. ✖ Krishna,80
Krishna,80

6406532239943. ✔ Siva,80
Krishna,80

6406532239944. ✖ Siva,50
Krishna,80

6406532239945. ✖ Krishna,50
Siva,80

Question Number : 144 Question Id : 640653668584 Question Type : MCQ Is Question

Mandatory : No Calculator : None Response Time : N.A Think Time : N.A Minimum Instruction Time : 0

Correct Marks : 7

Question Label : Multiple Choice Question

Consider the Java program below.

```

class A{
    void display(){
        System.out.println("Display A");
    }
}
class B extends A{
    void display(){
        super.display();
        System.out.println("Display B");
    }
    void display(String s){
        System.out.println("Display B." + s);
    }
}
class C extends B{
    void display(){
        super.display();
        System.out.println("Display C");
    }
    void display(String s){
        System.out.println("Display C." + s);
    }
}
class Example{
    public static void main(String[] args){
        B obj = new C(); // LINE 1
        obj.display();
        obj.display("Successful!"); // LINE 2
    }
}

```

Choose the correct option.

Options :

The program generates output:

Display A

Display B

Display C

6406532239950. ✓ Display C.Successful!

LINE 1 generates compilation error because a variable of type B cannot refer

6406532239951. ✗ to an object of type C.

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

```
Display B
Display C
Display C.Successful!
```

6406532239952. ✖

The program generates output:

```
Display A
Display B
Display B.Successful!
```

6406532239953. ✖

Question Number : 145 Question Id : 640653668586 Question Type : MCQ Is Question

Mandatory : No Calculator : None Response Time : N.A Think Time : N.A Minimum Instruction Time : 0

Correct Marks : 7

Question Label : Multiple Choice Question

Consider the code given below.

```

interface Teachable{
    public default void teaches(){
        System.out.println("Teaches");
    }
    public void manages();
}
abstract class Teacher implements Teachable{
    public void teaches() {
        System.out.println("Teach subjects");
    }
}
class HOD extends Teacher{
    public void manages() {
        System.out.println("Manages");
    }
}
public class Main {
    public static void main(String[] args) {
        Teachable obj1 = new HOD(); // LINE 1
        Teacher obj2 = new HOD();
        obj2.teaches();
        obj2.manages(); // LINE 2
    }
}

```

Choose the correct option.

Options :

This code generates the output:

Teach subjects

6406532239958. ✓ Manages

This code generates the output:

Teaches

6406532239959. ✗ Manages

LINE 1 generates compilation error because a variable of type Teachable

6406532239960. ✗ cannot refer to an object of type HOD.

LINE 2 generates compilation error because the method manages() cannot be

6406532239961. ✗ invoked on obj2

Question Number : 146 Question Id : 640653668588 Question Type : MCQ Is Question Mandatory : No Calculator : None Response Time : N.A Think Time : N.A Minimum Instruction Time : 0

Correct Marks : 7

Question Label : Multiple Choice Question

Consider the Java code given below.

```
class Employee{
    private String name;
    private double salary;
    public Employee(String n){
        name = n;
    }
    public Employee(double s){
        salary = s;
    }
    public Employee(String n, double s) {
        name = n;
        salary = s;
    }
    public String toString() {
        return "name = " + name + ", salary = " + salary;
    }
}
class Manager extends Employee{
    String dateOfPromotion;
    // ----- CODE BLOCK -----
    public String toString() {
        return super.toString()+" , dateOfPromotion = "+dateOfPromotion;
    }
}
```

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

Options :

```
public Manager(String dop) {
    dateOfPromotion = dop;
```

6406532239966. ✖ }

6406532239967. ✔

```
public Manager(String name, double salary, String dop) {  
    super(name, salary);  
    dateOfPromotion = dop;  
}
```

```
        public Manager(String name, double salary, String dop) {  
            name = name;  
            salary = salary;  
            dateOfPromotion = dop;  
6406532239968. ✖ }  
    }
```

```
        public Manager(String name, double salary, String dop) {  
            this.name = name;  
            this.salary = salary;  
            dateOfPromotion = dop;  
6406532239969. ✖ }  
    }
```

Sub-Section Number :	5
Sub-Section Id :	64065395182
Question Shuffling Allowed :	Yes
Is Section Default? :	null

Question Number : 147 Question Id : 640653668587 Question Type : MCQ Is Question Mandatory : No Calculator : None Response Time : N.A Think Time : N.A Minimum Instruction Time : 0

Correct Marks : 8

Question Label : Multiple Choice Question

Consider the code given below.

```

class Employee{
    private int id;
    private static double salary = 100;
    public Employee(int i){
        id = i;
    }
    public final double bonus(){
        return 0.1 * salary;
    }
}
class Manager extends Employee{
    public Manager(int x){
        super(x);
    }
    public final double bonus(){ // LINE 1
        return 0.2 * salary; // LINE 2
    }
}
public class Test{
    public static void main(String[] args){
        Employee e1 = new Manager(001); // LINE 3
        Manager m1 = new Employee(005); // LINE 4
        e1.bonus();
        m1.bonus();
    }
}

```

Which of the following statements is FALSE?

Options :

6406532239962. ✖ LINE 1 generates compilation error because the method `bonus()` cannot be overridden.

6406532239963. ✖ LINE 2 generates compilation error because instance variable `salary` cannot be accessed in class `Manager`.

6406532239964. ✔ LINE 3 generates compilation error because a variable of type `Employee` cannot refer to an object of type `Manager`.

6406532239965. ✖ LINE 4 generates compilation error because a variable of type `Manager` cannot refer to an object of type `Employee`.

Question Number : 148 Question Id : 640653668590 Question Type : MCQ Is Question Mandatory : No Calculator : None Response Time : N.A Think Time : N.A Minimum Instruction Time : 0

Correct Marks : 8

Question Label : Multiple Choice Question

Consider the code given below.

```

interface Iterator{
    public boolean has_next();
    public Object get_next();
}
abstract class Printable{
    public abstract void print();
}
class CustomerList{
    private class Customer extends Printable{
        private String name, billno;
        public Customer(String n, String b) {
            //initialize name and billno
        }
        public void print() {
            System.out.println(billno + ", " + name);
        }
    }
    private class CustIter implements Iterator{
        private int indx;
        public CustIter() {
            //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 CustIter();
    }
    private final int limit = 3;
    private Customer[] list = { new Customer("Abhishek", "C1001"),
        new Customer("Hari", "C1002"), new Customer("Sneha", "C1003")};
}
public class IterTest{
    public static void main(String[] args) {
        CustomerList cList = new CustomerList();
        Iterator iter = cList.getIterator();
        while(iter.has_next()) {
            _____;    //LINE 1
        }
    }
}

```

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

```

C1001, Abhishek
C1002, Hari
C1003, Sneha

```

Options :

6406532239974. ✓ ((Printable)iter.get_next()).print()

6406532239975. ✖ ((Customer)iter.get_next()).print()

6406532239976.

```
✖ ((CustomerList)iter.get_next()).print()
```

```
6406532239977. ✖ iter.get_next().print();
```

Sub-Section Number :	6
Sub-Section Id :	64065395183
Question Shuffling Allowed :	Yes
Is Section Default? :	null

Question Number : 149 Question Id : 640653668581 Question Type : MSQ Is Question Mandatory : No Calculator : None Response Time : N.A Think Time : N.A Minimum Instruction Time : 0

Correct Marks : 7 Max. Selectable Options : 0

Question Label : Multiple Select Question

Consider the code given below that checks whether two complex numbers are the same. Method `equals` is overridden to compare two `ComplexNumber` objects as follows. If two complex numbers have the same real part and imaginary part, then they are the same. Choose the correct option(s) to fill in place of CODE BLOCK so that the output is:
c1, c2 are same

```
class ComplexNumber{
    private int real; // real part
    private int img;  // imaginary part

    //Constructor to initialize instance variables

    public boolean equals(Object obj) {
        // CODE BLOCK
    }
}

public class Test {
    public static void main(String[] args) {
        ComplexNumber c1 = new ComplexNumber(3, 4);
        ComplexNumber c2 = new ComplexNumber(3, 4);
        if(c1.equals(c2))
            System.out.println("c1,c2 are same");
        else
            System.out.println("c1,c2 are different");

    }
}
```

Options :

```
if(obj instanceof ComplexNumber) {
    if(this.real == obj.real && this.img == obj.img)
        return true;
}
```

6406532239938. ✖ return false;

```
if(this.real == obj.real && this.img == obj.img)
    return true;
```

6406532239939. ✖ return false

```

        if(obj instanceof ComplexNumber) {
            ComplexNumber c = obj;
            if(this.real == c.real && this.img == c.img)
                return true;
        }
6406532239940. ✖ return false;

        if(obj instanceof ComplexNumber) {
            ComplexNumber c = (ComplexNumber) obj;
            if(this.real == c.real && this.img == c.img)
                return true;
        }
6406532239941. ✔ return false;

```

AppDev2

Section Id :	64065344904
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
Section Negative Marks :	0
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 :	64065395184
Question Shuffling Allowed :	No