6406531736874. ✖ $\frac{-4}{b^2-16}$

6406531736875. ✖ $\frac{8}{b^2-16}$

6406531736876. ✖ $\frac{-8}{b^2-16}$

Java

Section Id :	64065333936
Section Number :	8
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 :	64065373951
Question Shuffling Allowed :	No
Is Section Default? :	null

Question Number : 116 Question Id : 640653521106 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"

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 :

6406531736898. ✔ YES

6406531736899. ✖ NO

Sub-Section Number :	2
Sub-Section Id :	64065373952
Question Shuffling Allowed :	Yes
Is Section Default? :	null

Question Number : 117 Question Id : 640653521107 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.

```
abstract class CloudProviders{
    public abstract void storage();
}
class Azure extends CloudProviders{
    public void storage() {
         System.out.println("Azure storage");
    }
}
class AWS extends CloudProviders{
    public void storage() {
        System.out.println("AWS storage");
    }
}
class StorageList{
    private Object[] sArr = {new Azure(), new AWS()};
    public void getStorage(){
        for(int i = 0; i < sArr.length; i++){
            //LINE 1
        }
    }
}
public class Test{
    public static void main(String[] args) {
        StorageList sList = new StorageList();
        sList.getStorage();
    }
}
```

Identify the appropriate option to fill in place of `LINE 1` such that the output is

```
Azure storage
AWS storage
```

Options :

6406531736900. ✔ `((CloudProviders)sArr[i]).storage();`

6406531736901. ✖ `sArr[i].storage();`

6406531736902. ✖ `((Azure)sArr[i]).storage();`

6406531736903. ✖ `((AWS)sArr[i]).storage();`

Question Number : 118 Question Id : 640653521108 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 ArrayOperations{
    public <T> int countElement(T[] arr, T ele){
        // Counts the number of occurrences of ele in arr
    }
    public <T extends Comparable> void sort(T[] arr){
        // Sorts arr
   }
}
```

How does class `ArrayOperations` look after type erasure?

Options :

6406531736904. ✖

```
public class ArrayOperations{
    public int countElement(Object[] arr, Object ele){
        // Counts the number of occurrences of ele in arr
    }
    public void sort(Object[] arr){
        // Sorts arr
   }
}
```

6406531736905. ✔

```
public class ArrayOperations{
    public int countElement(Object[] arr, Object ele){
        // Counts the number of occurrences of ele in arr
    }
    public void sort(Comparable[] arr){
        // Sorts arr
   }
}
```

6406531736906. ✖

```
public class ArrayOperations{
    public int countElement(Object[] arr, Object ele){
        // Counts the number of occurrences of ele in arr
    }
    public void sort(T[] arr){
        // Sorts arr
   }
}
```

```
public class ArrayOperations{
    public int countElement(T[] arr, T ele){
        // Counts the number of occurrences of ele in arr
    }
    public void sort(T[] arr){
        // Sorts arr
   }
}
```

6406531736907.

Question Number : 119 Question Id : 640653521110 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 Employee {
    String name;
    // Constructor
    public String toString(){
        return name;
    }
}

class Manager extends Employee implements Cloneable {
    int nteam;
    // Constructor
    public Manager clone() throws CloneNotSupportedException{
        return (Manager)super.clone();
    }
    public String toString(){
        return (super.toString() + ": " + nteam);
    }
}
public class Test {
    public static void main(String[] args) throws CloneNotSupportedException{
        Manager m1 = new Manager("Hari", 4);
        Manager m2 = m1.clone();
        m2.name = "Reena";
        m2.nteam = 10;
        System.out.println(m1 + "\n" + m2);
    }
}
```

What will the output be?

Options :

6406531736912. ✖
```
Hari: 10
Reena: 10
```

6406531736913. ✔
```
Hari: 4
Reena: 10
```

6406531736914. ✖
```
Hari: 4
Reena: 4
```

6406531736915. ✖

```
Reena: 10
Reena: 10
```

Question Number : 120 Question Id : 640653521113 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 that takes as input the points obtained by teams in the matches they have played, and computes the total points obtained by each team. You may make use of the method description given below.

`getOrDefault(Object key, V defaultValue)`: Returns the value to which the specified `key` is mapped, or `defaultValue` if this map contains no mapping for the key.

```
import java.util.*;
class Team{
    String name, year;
    int points;
    // Constructor
}
public class MapTest{
    public static void printTeams(ArrayList<Team> tL) {
        var map = new LinkedHashMap<String, Integer>();
        Team tm = null;
        for(Team t:tL) {
            map.put(t.name, map.getOrDefault(t.name, 0)+t.points);
        }
        for (Map.Entry<String, Integer> e:map.entrySet()) {
            System.out.println(e.getKey()+" = "+e.getValue());
        }
    }
    public static void main(String[] args) {
        ArrayList<Team> tList = new ArrayList<Team>();
        tList.add(new Team("CSK", "2008", 14));
        tList.add(new Team("RCB", "2008", 8));
        tList.add(new Team("RCB", "2009", 14));
        tList.add(new Team("CSK", "2009", 12));
        printTeams(tList);
    }
}
```

What will the output be?

Options :

6406531736924. ✔
```
CSK = 26
RCB = 22
```

6406531736925. ✖
```
RCB = 22
CSK = 26
```

6406531736926. ✖
```
RCB = 14
CSK = 12
```

6406531736927. ✖
```
CSK = 12
RCB = 14
```

Question Number : 121 Question Id : 640653521114 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 following Java code.

```
import java.util.stream.*;
public class Test{
    public static void main(String[] args){
      Integer[] a = {12, 10, 13, 16};
      Stream.of(a)
            .map((i) -> i - 8).filter((i) -> i% 2 == 0)
            .forEach((x) -> System.out.println(x));
    }
}
```

What will the output be?

Options :

6406531736928. ✖
```
12
10
16
```

6406531736929. ✖ $\begin{matrix}4\\8\end{matrix}$

6406531736930. ✖ $\begin{matrix}12\\16\end{matrix}$

6406531736931. ✔ $\begin{matrix}4\\2\\8\end{matrix}$

Question Number : 122 Question Id : 640653521115 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 DaysException extends Exception{
    public DaysException(String str) {
        super(str);
    }
}
class HR{
    final static int no_of_hours = 20;
    public double getHoursPerDay(int no_of_days) {
        double hpd = 0.0;
        try {
            hpd = no_of_hours/no_of_days;
        }
        catch(ArithmeticException e) {
            e.initCause(new DaysException("No of days should not be 0"));
            throw e;
        }
        return hpd;
    }
}
public class ChainedException {
    public static void main(String[] args) {
        HR h = new HR();
        try {
            System.out.println(h.getHoursPerDay(0));
        }
        catch(ArithmeticException e) {
            System.out.println(e.getCause().getMessage());
        }
    }
}
```

Choose the correct option.

Options :

6406531736932. ✔ This program generates the output:
`No of days should not be 0`

6406531736933. ✖ This program generates the output:
`/ by zero`
`No of days should not be 0`

6406531736934. ✖

This program generates the output:

```
No of days should not be 0
/ by zero
```

This program generates the output:

```
/ by zero
```

6406531736935. ✖

Question Number : 123 Question Id : 640653521116 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.

```
import java.util.*;
public class QueueTest {
    public static void main(String[] args) {
        var queue = new ArrayDeque<Integer>();
        queue.add(23);
        queue.add(12);
        queue.add(43);
        while(queue.size() > 0) {
            System.out.println(queue.peek()+":"+queue.poll());
        }
    }
}
```

What will the output be?
You may make use of the descriptions of the methods given below. These are methods inside type Deque.
`poll()`: Retrieves and removes the head of the queue represented by this deque (in other words, the first element of this deque), or returns null if this deque is empty.
`peek()`: Retrieves, but does not remove, the head of the queue represented by this deque (in other words, the first element of this deque), or returns null if this deque is empty.

Options :

6406531736936. ✖

```
23:23
12:12
43:43
null:null
```

6406531736937. ✖
```
43:43
12:12
23:23
```

6406531736938. ✔
```
23:23
12:12
43:43
```

6406531736939. ✖
```
23:12
12:43
43:null
```

Question Number : 124 Question Id : 640653521117 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.

```
import java.util.*;
class Employee{
    String name;
    int no_of_leaves;
    //Constructor to initialize the instance variables
    //Method toString() to return name of the employee
}
public class IteratorTest {
    public static boolean property(int x) {
        if(x < 15)
            return true;
        return false;
    }
    public static void printAppraisalEmpList(List<Employee> eList){
        Iterator<Employee> it = eList.iterator();
        while (it.hasNext()) {
            Employee e = it.next();
            if(property(e.no_of_leaves))
                System.out.println(e);
            else
                it.remove(); // LINE 1
        }
    }
    public static void main(String[] args) {
        var list = new ArrayList<Employee>();
        list.add(new Employee("ABC", 15));
        list.add(new Employee("XYZ", 9));
        list.add(new Employee("PQR", 1));
        list.add(new Employee("MNO", 20));
        printAppraisalEmpList(list);
    }
}
```

Choose the correct option.

Options :

6406531736940. ✖ This program generates the output:
ABC
MNO

6406531736941. ✔ This program generates the output:
XYZ
PQR

6406531736942. ✖ LINE 1 generates the compilation error because method `remove()` is not defined in the `Iterator` interface.

6406531736943. ✖ LINE 1 should be replaced with `eList.remove(e);` to generate the output:

```
XYZ
PQR
```

Question Number : 125 Question Id : 640653521120 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 SetTest {
    public static void main(String[] args) {
        var set1 = new HashSet<String>();
        set1.add("India");
        set1.add("Sri Lanka");
        set1.add("Bangladesh");
        set1.add("Australia");
        var set2 = new TreeSet<String>(set1);

        Iterator<String> it1 = set1.iterator();
        Iterator<String> it2 = set2.iterator();

        while(it1.hasNext())
            System.out.println(it1.next());

        while(it2.hasNext())
            System.out.println(it2.next());
    }
}
```

Choose the correct option.

Options :

6406531736952. ✖ `it1` will visit elements of `set1` in sorted order.
`it2` will visit elements of `set2` in sorted order.

6406531736953. ✖ it1 will visit elements of set1 in the order in which they were inserted
it2 will visit elements of set2 in sorted order.

6406531736954. ✔ it1 will visit elements of set1 in unspecified order.
it2 will visit elements of set2 in sorted order.

6406531736955. ✖ it1 will visit elements of set1 in the order in which they were inserted.
it2 will visit elements of set2 in unspecified order.

Question Number : 126 Question Id : 640653521121 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

A. `throw`	I. Lists the types of exceptions that a method might throw
B. `throws`	II. Used to throw an exception object explicitly
C. `finally`	III. Executes only if exception is raised
D. `catch`	IV. Executes irrespective of whether the exception is raised or not

Options :

6406531736956. ✔
```
A---> II
B--> I
C--> IV
D--> III
```

6406531736957. ✖
```
A---> I
B--> II
C--> IV
D--> III
```

6406531736958. ✖
```
A---> II
B--> I
C--> III
D--> IV
```

A---> I
B--> II
C--> III
D--> IV

6406531736959.

Sub-Section Number : 3

Sub-Section Id : 64065373953

Question Shuffling Allowed : Yes

Is Section Default? : null

Question Number : 127 Question Id : 640653521109 Question Type : MSQ Is Question Mandatory : No Calculator : None Response Time : N.A Think Time : N.A Minimum Instruction Time : 0

Correct Marks : 4 Selectable Option : 0

Question Label : Multiple Select Question

Consider the Java code given below that prints the sum of elements of a list. From among the options, identify the appropriate function header for function elementSum that takes as input a list of numbers, and prints the sum of the elements of the list.

```
import java.util.*;
class Test {
    // FUNCTION HEADER for function elementSum
    {
        // Prints the sum of elements of list
    }
    public static void main(String[] args) {
        List<Integer> l = new ArrayList<>();
        l.add(12);
        l.add(23);

        List<Float> l1 = new ArrayList<>();
        l1.add(12.2f);
        l1.add(23.4f);

        elementSum(l);
        elementSum(l1);
    }
}
```

Choose the correct option(s).

Options :

6406531736908. ✖ `public static void elementSum(List<Number> lst)`

6406531736909. ✔ `public static <T extends Number> void elementSum(List<T> lst)`

6406531736910. ✔ `public static void elementSum(List<? extends Number> lst)`

6406531736911. ✖ `public static void elementSum(List<Double> lst)`

Question Number : 128 Question Id : 640653521111 Question Type : MSQ Is Question Mandatory : No Calculator : None Response Time : N.A Think Time : N.A Minimum Instruction Time : 0

Correct Marks : 4 Selectable Option : 0

Question Label : Multiple Select Question

Consider the Java code given below that prints the highest tax among a set of given `Taxable` objects. From among the options, identify the appropriate function header for function `printTax` that takes as input an array of `Taxable` objects and prints the highest tax.

```
import java.util.*;
interface Taxable {
    public abstract double findTax();
}
class Employee implements Taxable{
    double salary;
    // Constructor
    // method findTax() that returns tax of Employee which is 10% of salary
}
class Manager extends Employee{
    // Constructor
}
public class Test{
    // LINE 1: FUNCTION HEADER
    {
        // invokes method findTax()
        // to print the value of highest tax
    }

    public static void main(String[] args) {
        Taxable[] t = {new Employee(400), new Manager(3000)};
        printTax(t);
    }
}
```

Choose the correct option(s).

Options :

6406531736916. ✖ `public static void printTax(<?> t)`

6406531736917. ✔ `public static <T extends Taxable> void printTax(T[] c)`

6406531736918. ✖ `public static <T extends Manager> void printTax(T[] c)`

6406531736919. ✔ `public static void printTax(Taxable[] c)`

Question Number : 129 Question Id : 640653521112 Question Type : MSQ Is Question Mandatory : No Calculator : None Response Time : N.A Think Time : N.A Minimum Instruction Time : 0

Correct Marks : 4 Selectable Option : 0

Question Label : Multiple Select Question

Consider the Java code given below that should print the names of students whose total is between 255 and 360 (both inclusive).

```
import java.util.*;
class Student{
    String name;
    double total;
    public Student(String name, double total) {
        this.name = name;
        this.total = total;
    }
}
public class Test {
    public static void main(String[] args) {
        List<Student> sList = new ArrayList<Student>();
        sList.add(new Student("s1", 360));
        sList.add(new Student("s2", 400));
        sList.add(new Student("s3", 200));
        sList.add(new Student("s4", 255));
        //CODE BLOCK
    }
}
```

Choose the correct option(s) to fill in place of `CODE BLOCK` to obtain the right answer.

Options :

6406531736920. ✖

```
sList.stream()
    .map(i -> i.total >= 255 && i.total <= 360)
    .forEach(s->System.out.println(s.name));
```

6406531736921. ✔

```
sList.stream()
    .filter(i -> i.total >= 255 && i.total <= 360)
    .forEach(s->System.out.println(s.name));
```

6406531736922. ✔

```
sList.stream()
    .filter(i -> i.total >= 255)
    .filter(i -> i.total <= 360)
    .forEach(s->System.out.println(s.name));
```

```
sList.stream()
    .filter(i -> i.total >= 255)
    .map(i -> i.total <= 360)
    .forEach(s->System.out.println(s.name));
```

6406531736923. ✖

Sub-Section Number : 4

Sub-Section Id : 64065373954

Question Shuffling Allowed : Yes

Is Section Default? : null

Question Number : 130 Question Id : 640653521118 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.

```
class Customer{
    private String name, aadhar, pan;

    //Constructor to initialize the instance variables

    public void getDetails() {
        String msg = ""PAN/Aadhar required"";
        System.out.println(name);
        assert aadhar.length() == 12 || pan.length() == 10: msg; //LINE 1
        if(aadhar.length() == 12)
            System.out.println(aadhar);
        if(pan.length() == 10)
            System.out.println(pan);
    }
}
public class AssertionTest {
    public static void main(String[] args) {
        Customer c1 = new Customer("Shreyas Iyer", "","BXBPB1123D");
        Customer c2 = new Customer("Venkatesh Iyer", "209005091129","");
        c1.getDetails(); // LINE 2
        c2.getDetails(); // LINE 3
    }
}
```

Choose the correct option when the program is executed as:
`java -ea AssertionTest`

Options :

6406531736944. ✖ LINE 1 generates a compilation error, because you cannot write multiple conditions in a single assert statement.

6406531736945. ✖ LINE 1 throws AssertionError when LINE 2 is executed.

6406531736946. ✖ LINE 1 throws AssertionError when LINE 3 is executed.

6406531736947. ✔ This program generates the output:

```
Shreyas Iyer
BXBPB1123D
Venkatesh Iyer
209005091129
```

Question Number : 131 Question Id : 640653521119 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.

```
import java.util.*;
class LocationException extends Exception{
    public LocationException(String str) {
        super(str);
    }
}
class Fresher{
    String name, pre_location;
    //Constructor to initialize the instance variables
}
class Kipro{
    Map<String, String> map = new HashMap<String, String>();
    public Kipro() {
        map.put("Hyderabad", "100 fresher jobs");
        map.put("Chennai", "175 fresher jobs");
    }
    void recruit(Fresher f) throws LocationException {
        if(map.get(f.pre_location) == null)
            throw new LocationException("No jobs");
        else
            System.out.println(map.get(f.pre_location));
    }
}
public class ExceptionTest {
    public static void main(String[] args) {
        Fresher f1 = new Fresher("ABC", "Hyderabad");
        Fresher f2 = new Fresher("XYZ", "Bangalore");
        Kipro k = new Kipro();
        try {
            k.recruit(f1);
            k.recruit(f2);
        }
        catch (LocationException e) {
            System.out.println(e.getMessage());
        }
    }
}
```

Choose the correct option.

Options :

6406531736948. ✖

This program generates the output:

```
No jobs
100 fresher jobs
```

6406531736949. ✔

This program generates the output:

```
100 fresher jobs
No jobs
```

6406531736950. ✖

This program generates the output:

```
No jobs
175 fresher jobs
```

6406531736951. ✖

The program terminates due to unhandled exception(s).

AppDev2

Section Id :	64065333937
Section Number :	9
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 :	64065373955