The objective function for the problem is

Options :

6406531484725. ✗ min 50*x* + 15*y*

6406531484726. ✓ max 50*x* + 15*y*

6406531484727. ✗ min 50*x* − 15*y*

6406531484728. ✗ max 50*x* − 15*y*

Question Number : 152 Question Id : 640653445595 Question Type : MSQ Is Question Mandatory : No Calculator : None Response Time : N.A Think Time : N.A Minimum Instruction Time : 0

Correct Marks : 3 Selectable Option : 0

Question Label : Multiple Select Question

The constraint function for the problem is

Options :

6406531484729. ✓ $x \geq 0, y \geq 0$

6406531484730. ✓ $5x + 2y \leq 100$

6406531484731. ✗ $5x + 2y \geq 100$

6406531484732. ✓ $x + y \leq 60$

Java

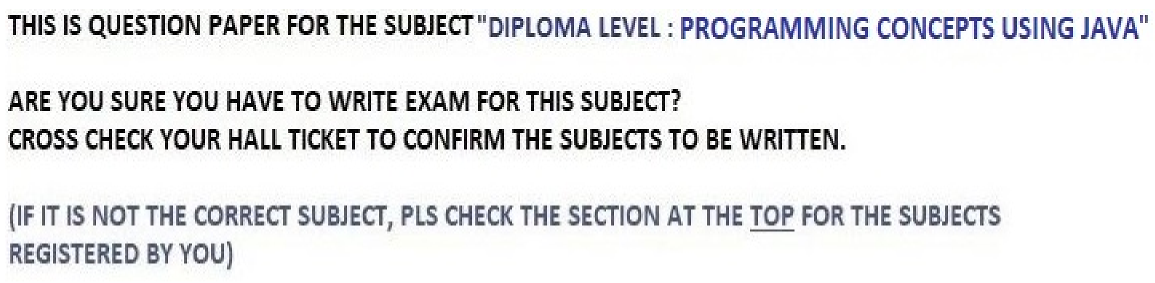
Section Id :	64065328983
Section Number :	9
Section type :	Online
Mandatory or Optional :	Mandatory
Number of Questions :	16

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 :	64065363332
Question Shuffling Allowed :	No
Is Section Default? :	null

Question Number : 153 Question Id : 640653445596 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 :

6406531484733. ✓ YES

6406531484734. ✗ NO

Sub-Section Number :	2
Sub-Section Id :	64065363333
Question Shuffling Allowed :	Yes
Is Section Default? :	null

Question Number : 154 Question Id : 640653445597 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 SmartSpeaker{
    public abstract void output();
}
class Echo extends SmartSpeaker{
    public void output() {
         System.out.println("Echo speaks");
    }
}
class EchoDot extends SmartSpeaker{
    public void output() {
        System.out.println("EchoDot speaks");
    }
}
class DeviceList{
    private Object[] dArr = {new Echo(), new EchoDot()};
    public void getOutput(){
      for(int i = 0; i < dArr.length; i++){
        //LINE 1
      }
    }
}
public class Test{
    public static void main(String[] args) {
        DeviceList l = new DeviceList();
        l.getOutput();
    }
}
```

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

```
Echo speaks
EchoDot speaks
```

Options :

6406531484735. ✖ `dArr[i].output();`

6406531484736. ✖ `((EchoDot)dArr[i]).output();`

6406531484737. ✖ `((Echo)dArr[i]).output();`

6406531484738. ✔ `((SmartSpeaker)dArr[i]).output();`

Question Number : 155 Question Id : 640653445598 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 ArrayExample{
   public <T extends Comparable> void sortArray(T[] obj){
     // Sorts obj
   }
   public <T> void elementDisplay(T[] arr){
      System.out.println(arr[0]);
   }
}
```

How does class ArrayExample look after type erasure?

Options :

```
public class ArrayExample{
   public void sortArray(Object[] obj){
        //Sorts obj
   }
   public void elementDisplay(Object[] arr){
      System.out.println(arr[0]);
   }
}
```

6406531484739. ✖

```
public class ArrayExample{
   public void sortArray(Comparable[] obj){
        //Sorts obj
   }
   public void elementDisplay(T[] arr){
      System.out.println(arr[0]);
   }
}
```

6406531484740. ✖

```
public class ArrayExample{
   public void sortArray(Comparable[] obj){
        //Sorts obj
   }
   public void elementDisplay(Object[] arr){
      System.out.println(arr[0]);
   }
}
```

6406531484741. ✓

```
public class ArrayExample{
   public void sortArray(T[] obj){
        //Sorts obj
   }
   public void elementDisplay(T[] arr){
      System.out.println(arr[0]);
   }
}
```

6406531484742. ✗

Question Number : 156 Question Id : 640653445599 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 returns `true` if the runs scored by at least one non-captain player is more than that of the runs scored by the captain. From among the options, identify the appropriate function header for function `findPlayer` that takes as input an array of `Player` objects and a `Captain` object, and finds if such a player exists.

```
import java.util.*;
interface Playable{
    public abstract int getScore();
}
class Player implements Playable{
    private String name;
    private int runs;
    /---- Constructor
    /---- Accessor methods
    public int getScore() {
      return runs;
    }
}
class Captain extends Player{
    /---- Constructor
}
public class Test{
    // FUNCTION HEADER for function findPlayer
    {
        /------ returns true if at least one player has scored
              more than the captain, else false
    }
    public static void main(String[] args) {
        Player[] p = {new Player("ABC",23), new Player("XYZ",40)};
        Captain c = new Captain("CAP",34);
        System.out.println(findPlayer(p,c));
    }
}
```

Choose the correct option.

Options :

6406531484743. ✖ `public static <T, S extends T> boolean findPlayer(T[] arr, S c)`

6406531484744. ✓ `public static <T extends Playable, S extends T> boolean findPlayer(T[] arr, S c)`

6406531484745. ✖ `public static <T extends S, S> boolean findPlayer(T[] arr, S c)`

6406531484746. ✖ `public static <T extends S, S extends Playable> boolean findPlayer(T[] arr, S c)`

Question Number : 157 Question Id : 640653445601 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 AgeLimitException extends Exception{
    public AgeLimitException() {
        System.out.println("Age should be atleast 18");
    }
}
public class FClass{
    public static void castVote(int age) throws AgeLimitException {
        if(age < 18)
            throw new AgeLimitException();
        System.out.println("Voted successfully");
    }
    public static void main(String[] args) {
        try {
            castVote(10);
            castVote(20);             // LINE 1
        }
        catch(AgeLimitException ae) {
            System.out.println("Cannot vote");
        }
        catch(Exception e) {          // LINE 2

        }
    }
}
```

Choose the correct option.

Options :

6406531484751. ✖ LINE 1 generates compilation error because of unreachable code

6406531484752. ✖

This program generates output:

```
Age should be atleast 18
Cannot vote
Voted successfully
```

6406531484753. ✓ This program generates output:

```
Age should be atleast 18
Cannot vote
```

6406531484754. ✗ LINE 2 generates compilation error because of unreachable code

Question Number : 158 Question Id : 640653445602 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 Example{
    List<Integer> lst;
    public Example(List<Integer> l){
      lst = l;
    }
    public void copyList(int[] arr){
        try {
            for(int i = 0; i < lst.size(); i++){
              arr[i] = lst.get(i);
              System.out.println(arr[i]);
            }
        }
        catch(ArrayIndexOutOfBoundsException ae) {
          System.out.println("cannot copy");
        }
    }
}
 public class Test {
     public static void main(String[] args){
         List<Integer> l = new ArrayList<>();
         l.add(10);
         l.add(20);
         l.add(30);
         int[] a = new int[2];
         Example e1 = new Example(l);
         try {
             e1.copyList(a);
         }
         catch(Exception e) {
             System.out.println("caught in main");
         }
     }
 }
```

Options :

6406531484755. ✗ This program generates the output:

```
10
20
cannot copy
caught in main
```

6406531484756. ✓

This program generates the output:

```
10
20
cannot copy
```

6406531484757. ✖ This program generates the output:

```
10
20
30
```

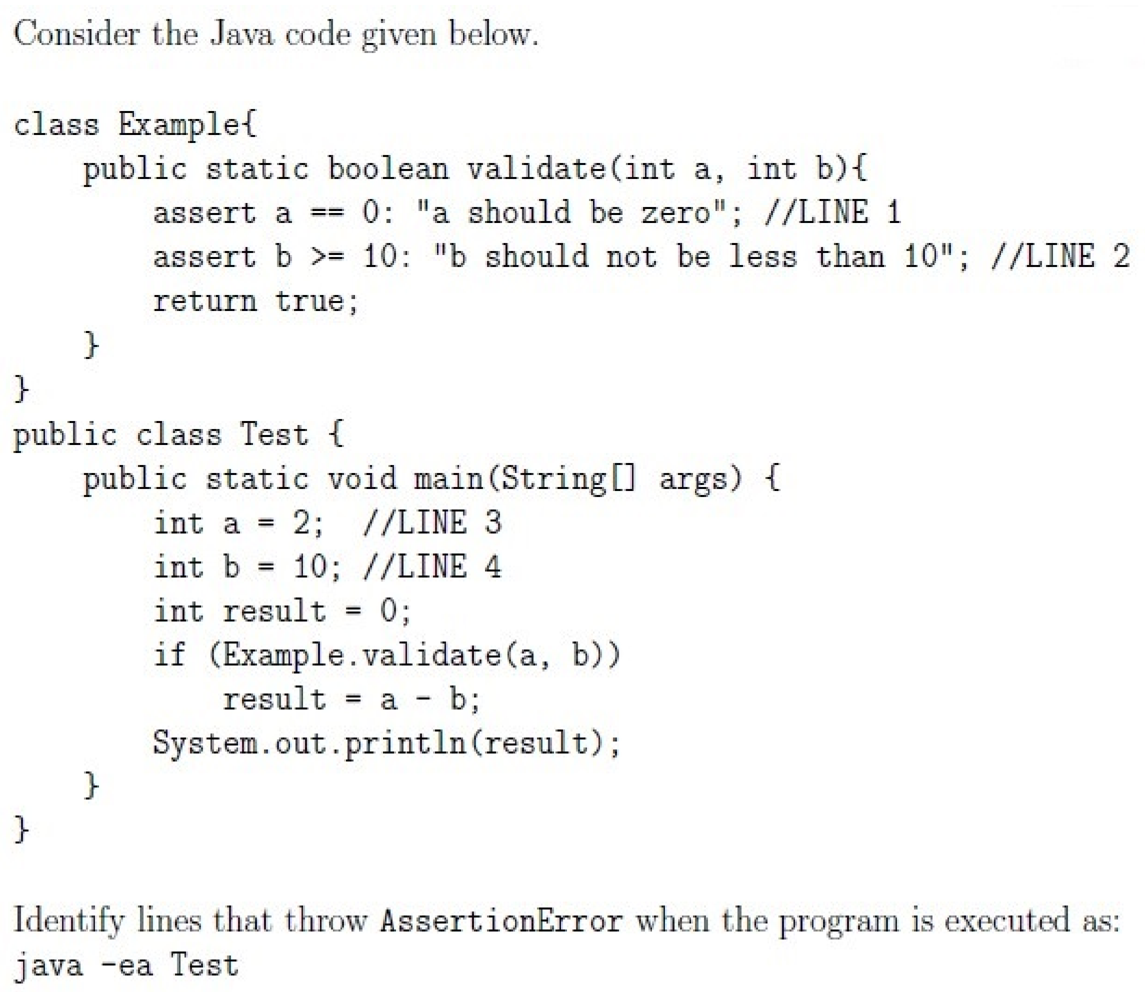
6406531484758. ✖ This program generates the output:

```
cannot copy
caught in main
```

Question Number : 159 Question Id : 640653445603 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 Example{
    public static boolean validate(int a, int b){
        assert a == 0: "a should be zero"; //LINE 1
        assert b >= 10: "b should not be less than 10"; //LINE 2
        return true;
    }
}
public class Test {
    public static void main(String[] args) {
        int a = 2;  //LINE 3
        int b = 10; //LINE 4
        int result = 0;
        if (Example.validate(a, b))
            result = a - b;
        System.out.println(result);
    }
}
```

Identify lines that throw `AssertionError` when the program is executed as:
`java -ea Test`

Options :

6406531484759. ✓ LINE 1

6406531484760. ✖ LINE 2

6406531484761. ✖ LINE 3

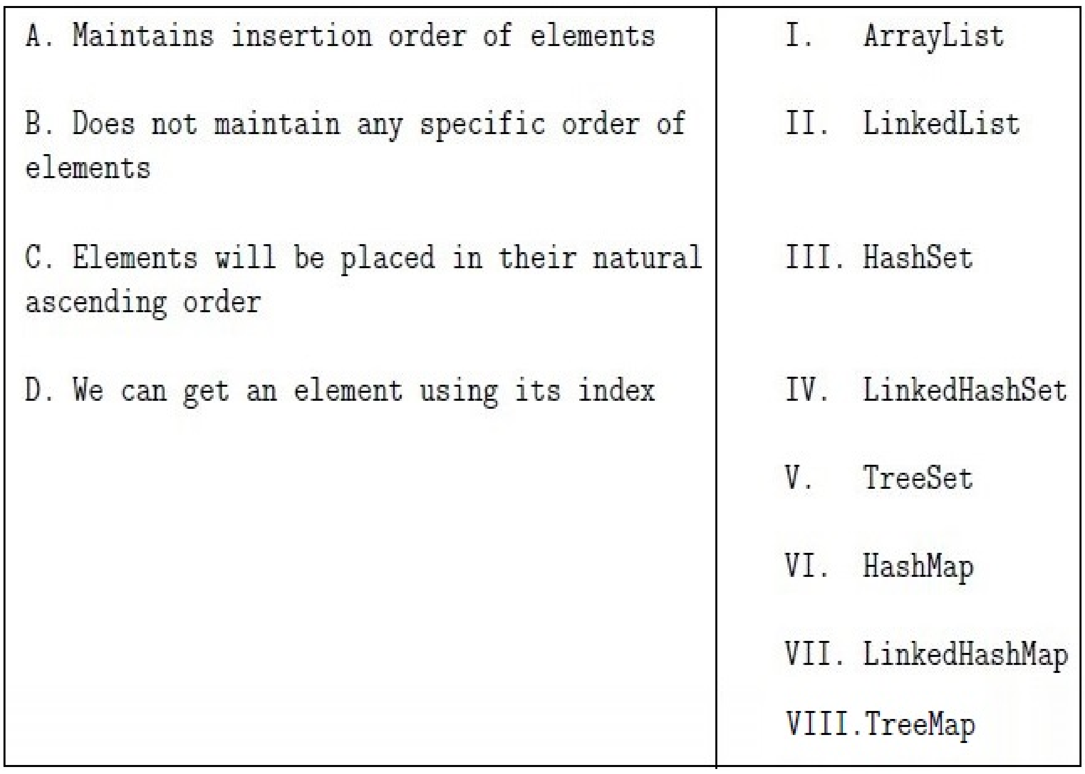
6406531484762. ✖ LINE 4

Question Number : 160 Question Id : 640653445605 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. Maintains insertion order of elements	I. ArrayList
B. Does not maintain any specific order of elements	II. LinkedList
C. Elements will be placed in their natural ascending order	III. HashSet
D. We can get an element using its index	IV. LinkedHashSet
	V. TreeSet
	VI. HashMap
	VII. LinkedHashMap
	VIII.TreeMap

Options :

6406531484767. ✖
```
A---> IV, VII
B--> III, VI
C--> V, VIII
D--> I, II
```

6406531484768. ✖
```
A---> I, II, V, VIII
B--> III, VI
C--> IV, VII
D--> I, II
```

6406531484769. ✔
```
A---> I, II, IV, VII
B--> III, VI
C--> V, VIII
D--> I, II
```

6406531484770. ✖
```
A---> V, VIII
B--> III, VI
C--> I, II, IV, VII
D--> I, II
```

Question Number : 161 Question Id : 640653445606 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 checks whether the input string is a palindrome or not.

```
import java.util.*;
public class QTest {
    public static boolean check(Deque<Character> q) {
        //CODE BLOCK
        return q.isEmpty();
    }
    public static void main(String[] args) {
        String str1 = "HANNAH";
        String str2 = "BANANA";
        Deque<Character> queue1 = new ArrayDeque<Character>();
        Deque<Character> queue2 = new ArrayDeque<Character>();
        for(int i = 0 ;i < 6; i++) {
            queue1.add(str1.charAt(i));
            queue2.add(str2.charAt(i));
        }
        System.out.println(check(queue1));
        System.out.println(check(queue2));
    }
}
```

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

```
true
false
```

You may make use of the descriptions of the methods given below. These are methods inside type Deque.
`pollLast()`: Retrieves and removes the last element of this deque, or returns null if this deque is empty.
`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.
`isEmpty()`: Returns true if this deque contains no elements.

Options :

6406531484771. ✖

```
while(q.size() < 0) {
    while(q.poll() != q.pollLast())
        break;
}
```

```
while(q.size() < 0) {
    if(q.poll() != q.pollLast())
        break;
}
```

6406531484772. ✖

```
while(q.size() > 0) {
    if(q.poll() != q.pollLast())
        break;
}
```

6406531484773. ✔

```
while(q.size() > 0) {
    while(q.poll() != q.pollLast())
        break;
}
```

6406531484774. ✖

Question Number : 162 Question Id : 640653445607 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 :

6406531484775. ✓
```
CSK = 26
RCB = 22
```

6406531484776. ✗
```
RCB = 22
CSK = 26
```

6406531484777. ✗
```
RCB = 14
CSK = 12
```

6406531484778. ✖ CSK = 12
RCB = 14

Sub-Section Number : 3

Sub-Section Id : 64065363334

Question Shuffling Allowed : Yes

Is Section Default? : null

Question Number : 163 Question Id : 640653445600 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 three Java files given below.

Calculator.java:

```
package pack1;
public class Calculator {
    public int sum(int a, int b) {
        return a + b;
    }
    protected int div(int a, int b) {
        return a / b;
    }
    private int diff(int a, int b) {
        return a - b;
    }
}
```

SmartCalculator.java:

```
package pack1;
public class SmartCalculator extends Calculator{
    public void time() {
        System.out.println("prints current time");
    }
}
```

Test.java:

```
package pack1;
public class Test {
    public static void main(String[] args) {
        Calculator c1 = new SmartCalculator();
        System.out.println(c1.div(10, 2));     //LINE 1
        System.out.println(c1.time());         //LINE 2
        System.out.println(c1.sum(10, 20));    //LINE 3
        System.out.println(c1.diff(10, 20));   //LINE 4
    }
}
```

Choose the correct option.

Options :

6406531484747. ✖ This program generates the output:

```
5
prints current time
30
-10
```

6406531484748. ✖ LINE 1 and LINE 2 generate compilation errors.

6406531484749. ✗ LINE 1 and LINE 4 generate compilation errors.

6406531484750. ✓ LINE 2 and LINE 4 generate compilation errors.

Question Number : 164 Question Id : 640653445604 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 Player implements Cloneable{
    String name;
    ArrayList<String> achievements = new ArrayList<String>();
    public Player(String n, ArrayList<String> a){
        name = n;
        achievements = a;
    }
    public Object clone() throws CloneNotSupportedException{
        Player p = (Player) super.clone();
        ArrayList<String> achobj = new ArrayList<String>();
        for (String str: this.achievements)
            achobj.add(str);
        p.achievements = achobj;
        return p;
    }
    public String toString(){
        return name+": "+achievements;
    }
}
public class CloTest{
    public static void main(String[] args){
        ArrayList<String> pa = new ArrayList<String>();
        pa.add("MoM(10 times)");            //MoM: Man of the Match
        pa.add("PoS(5 times)");             //PoS: Player of the Series
        Player p1 = new Player("Rohit", pa);
        try{
            Player p2 = (Player)p1.clone();
            p1.achievements.add("WR");      //WR: World Record
            p2.name = "Kohli";
            System.out.println(p1+"\n"+p2);
        }
        catch(CloneNotSupportedException e){
            System.out.println(e);
        }
    }
}
```

What will the output be?

Options :

6406531484763. ✖
```
Kohli: [MoM(10 times), PoS(5 times), WR]
Kohli: [MoM(10 times), PoS(5 times), WR]
```

6406531484764. ✖
```
Rohit: [MoM(10 times), PoS(5 times), WR]
Kohli: [MoM(10 times), PoS(5 times), WR]
```

6406531484765. ✓

```
Rohit:  [MoM(10 times), PoS(5 times), WR]
Kohli:  [MoM(10 times), PoS(5 times)]
```

6406531484766. ✖

```
Rohit:  [MoM(10 times), PoS(5 times)]
Kohli:  [MoM(10 times), PoS(5 times), WR]
```

Sub-Section Number :	4
Sub-Section Id :	64065363335
Question Shuffling Allowed :	Yes
Is Section Default? :	null

Question Number : 165 Question Id : 640653445608 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 following program.

```
class Point<T extends Number>{
    private T x;
    private T y;
    public Point(T x_val, T y_val){
        x = x_val;
        y = y_val;
    }
    public T get_x() {
        return x;
    }
    public T get_y() {
        return y;
    }
    public Point<Double> addition(____LINE 1____) {
        Point<Double> p1 = new Point<>(0.0, 0.0);
        p1.x = this.x.doubleValue() + p.get_x().doubleValue();
        p1.y = this.y.doubleValue() + p.get_y().doubleValue();
        return p1;
    }

}
public class Test{
    public static void main(String args[]) {
        Point<Integer> p1 = new Point<Integer>(3, 4);
        Point<Integer> p2 = new Point<Integer>(3, 2);
        Point<Double> p3 = p1.addition(p2);
        System.out.println(p3.get_x() + "," + p3.get_y() );
    }
}
```

Choose all the options which can be used in place of `LINE 1` for a successful compilation of the given code.

Options :

6406531484779. ✗ `Point<Number> p`

6406531484780. ✓ `Point<T> p`

6406531484781. ✗ `Point<Object> p`

6406531484782. ✓ `Point<?> p`

Question Number : 166 Question Id : 640653445610 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 names of consumers whose CIBIL score (which gives a summary of one's credit history) is between 500 and 700 (both including).

```
import java.util.*;
class Consumer{
    String name;
    double cibilscore;
    public Consumer(String name, double cibilscore) {
        this.name = name;
        this.cibilscore = cibilscore;
    }
}
public class StreamEx {
    public static void main(String[] args) {
        List<Consumer> list = new ArrayList<Consumer>();
        list.add(new Consumer("ABC", 560));
        list.add(new Consumer("PQR", 750));
        list.add(new Consumer("MNO", 400));
        list.add(new Consumer("XYZ", 350));
        list.add(new Consumer("EFG", 578));
        //CODE BLOCK
    }
}
```

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

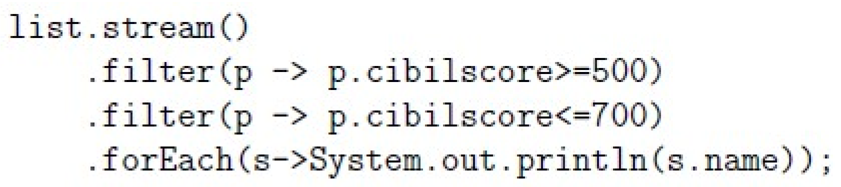
Options :

6406531484787. ✖

```
list.stream()
    .map(p -> p.cibilscore>=500 && p.cibilscore<=700)
    .forEach(s->System.out.println(s.name));
```

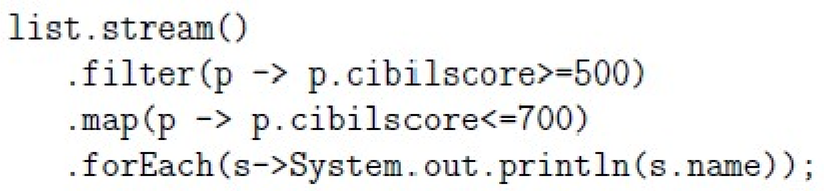
6406531484788. ✓

```
list.stream()
    .filter(p -> p.cibilscore>=500 && p.cibilscore<=700)
    .forEach(s->System.out.println(s.name));
```



```
list.stream()
    .filter(p -> p.cibilscore>=500)
    .filter(p -> p.cibilscore<=700)
    .forEach(s->System.out.println(s.name));
```

6406531484789. ✓



```
list.stream()
   .filter(p -> p.cibilscore>=500)
   .map(p -> p.cibilscore<=700)
   .forEach(s->System.out.println(s.name));
```

6406531484790. ✗

Question Number : 167 Question Id : 640653445611 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.

```
import java.util.*;
class Player{
    String name;
    int runs;
    public Player(String n, int r) {
        name = n;
        runs = r;
    }
    public String toString() {
        return "name=" + name + ", runs=" + runs;
    }
}
public class Main {
    public static void printTopBatsman(List<Player> p){
        //  CODE BLOCK
    }
    public static void main(String[] args) {
        var pList=new ArrayList<Player>();
        pList.add(new Player("Shami", 579));
        pList.add(new Player("Bumrah", 450));
        pList.add(new Player("Rohit", 1700));
        pList.add(new Player("Kohli", 1850));
        printTopBatsman(pList);
    }
}
```

Choose the correct option(s) to fill in place of `CODE BLOCK` so that the program outputs the details about the batsman who has scored maximum runs:

```
name=Kohli, runs=1850
```

Options :

6406531484791. ✗

```
Collections.sort(p);
p.stream().limit(1).forEach(System.out::println);
```

6406531484792. ✗

```
Collections.sort(p);
p.stream().forEach(System.out::println);
```

6406531484793. ✓

```
Collections.sort(p, (r2, r1)-> r1.runs-r2.runs);
p.stream().limit(1).forEach(System.out::println);
```

6406531484794. ✓

```
Collections.sort(p, (r1, r2)-> r2.runs-r1.runs);
p.stream().limit(1).forEach(s->System.out.println(s));
```

Sub-Section Number : 5

Sub-Section Id : 64065363336

Question Shuffling Allowed : Yes

Is Section Default? : null

Question Number : 168 Question Id : 640653445609 Question Type : MSQ Is Question Mandatory : No Calculator : None Response Time : N.A Think Time : N.A Minimum Instruction Time : 0

Correct Marks : 3 Selectable Option : 0

Question Label : Multiple Select Question

Consider the Java code given below.

```
interface Queue1<E>{
    public void add();
    public E poll();
}
interface Queue2<E> extends Queue1<E>{
    public void addFirst();
    public E pollLast();
}
class QueueImpl1<E> implements Queue1<E>{
    //implemented abstract methods
}
class QueueImpl2<E> implements Queue2<E>{
    //implemented abstract methods
}
public class IndirectionTest {
    public static void main(String[] args) {
        //CODE BLOCK
    }
}
```

Identify the correct option to be filled in place of CODE BLOCK to create objects for QueueImpl1 and QueueImpl2.

Options :

6406531484783. ✖

```
Queue2<String> q1, q2;
q1 = new QueueImpl1<String>();
q2 = new QueueImpl2<String>();
```

6406531484784. ✓

```
Queue1<String> q1, q2;
q1 = new QueueImpl1<String>();
q2 = new QueueImpl2<String>();
```

6406531484785. ✖

```
QueueImpl1<String> q1, q2;
q1 = new QueueImpl1<String>();
q2 = new QueueImpl2<String>();
```

6406531484786. ✖

```
QueueImpl2<String> q1, q2;
q1 = new QueueImpl1<String>();
q2 = new QueueImpl2<String>();
```

AppDev2

Section Id :	64065328984
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 :	64065363337