Focus Area 1

1 2 3 4 5

[2 Marks]

Question 2

Predict the output of the below code snippet:

```
public class SetDemo {
        public static void main(String[] args) {
                Set<String> set = new HashSet<>();
                set.add("A");
                set.add("B");
                set.add("C");
                set.add("D");
                set.add("B");
                Iterator<String> iterator = set.iterator();
                while(iterator.hasNext()) {
                        if(iterator.next() == "B") {
                                set.remove(iterator.next());
                                System.out.println(iterator.next());
                        }
                }
        }
}
```

33%

Assumption:

Focus Area 1

1 2 3 4 5

```
                if(iterator.next() == "B") {
                    set.remove(iterator.next());
                    System.out.println(iterator.next());
                }
            }
        }
}
```

Assumption:

1. All necessary imports are done

- C
- D
- B
- ConcurrentModificationException exception will be thrown

Reset Save

Focus Area 1

2

4

}

}

Assumption:

1. All necessary imports are done

101- Blowing with the Wind
102- Stand By Me
null- Annie's Song
104- Here I Am

101- Blowing with the Wind
102- Stand By Me

NullPointerException exception will be thrown

null- Annie's Song
101- Blowing with the Wind
102- Stand By Me
104- Here I Am

Reset

Save

Focus Area 1

2

4

Question 3

What will be the output of the below code snippet?

```
public class MapTester {
        public static void main(String[] args) {
                Map<Integer, String> songMap = new TreeMap<Integer, String>();
                songMap.put(101, "Blowing in the Wind");
                songMap.put(102, "Stand By Me");
                songMap.put(null, "Annie's Song");
                songMap.put(104, "Here I Am");
                for(Map.Entry<Integer, String> ele: songMap.entrySet()) {
                        System.out.println(ele.getKey()+"- "+ele.getValue());
                }
        }
}
```

51% 49%

Assumption:

1. All necessary imports are done

101- Blowing with the Wind

102- Stand By Me

OBJECTIVE QUESTIONS

Focus Area 1

1 2

3 4

5

on Number 5

Question 4

72%

[1 Mark]

Assuming the below code executes in the main method. What will be the output of the following code?

```
LocalDate localDate = LocalDate.of(2012, 12, 31);
localDate=localDate.plus(2, ChronoUnit.MONTHS);
System.out.println(localDate);
```

- 2013-03-02
- 2013-02-28
- 2013-03-01
- Error: Cannot add month

Reset Save

Focus Area 1

1 2 3 4 5

```
        Customer c3 = new Customer(9003, "Anupam");
        list.add(c1);
        list.add(c2);
        list.add(c3);
        Stream<Customer> stream = list.stream().filter(cust -> cust.getCustomerName().length() >=
        stream.forEach((customer) -> System.out.print(customer.getCustomerId() + "  "));
    }
}
```

- 9001
- 9001 9003
- 9001 9002 9003
- 9002 9003

Reset Save

Focus Area 1

1 2 3 4 5

What will be the output of the below code? Assume **Customer class** exists as below

83% 17%

```
class Customer{
   int customerId;
   String customerName;
   Customer(int cId,String name){
       this.customerId=cId;
       this.customerName=name;
   }
   //Getters and setters
}
public class Tester2 {
    public static void main(String a[]) {
        List<Customer> list = new ArrayList<Customer>();
        Customer c1 = new Customer(9001, "Patrick");
        Customer c2 = new Customer(9002, "Mary");
        Customer c3 = new Customer(9003, "Anupam");
        list.add(c1);
        list.add(c2);
        list.add(c3);
        Stream<Customer> stream = list.stream().filter(cust -> cust.getCustomerName().length() >=
        stream.forEach((customer) -> System.out.print(customer.getCustomerId() + "  "));
    }
}
```

02 : 57
Hours Minutes

OBJECTIVE Q

Focus Area 1

1 2 3 4 5

[1 Mark]

What will be the output of the below program?

```
public class Tester{
  public static void main(String[] args) {
    Map<String, String> map1 = new LinkedHashMap<>();
    Map<String, String> map2 = new LinkedHashMap<>();
    map1.put("1001", "Jack");
    map1.put("1002", "Peter");
    map2.put("1002", "1002");
    map1.putAll(map2);
    Set<Entry<String,String>> set1 = map1.entrySet();
    for (Entry<String, String> entry : set1) {
        System.out.println(entry.getValue());
    }
  }
}
```

Note: Assume the required imports are added.

- Peter 1001
- Jack 1002

02 : 47 : 59
Hours Minutes Seconds

OBJECTIVE QUESTION

ocus Area 1

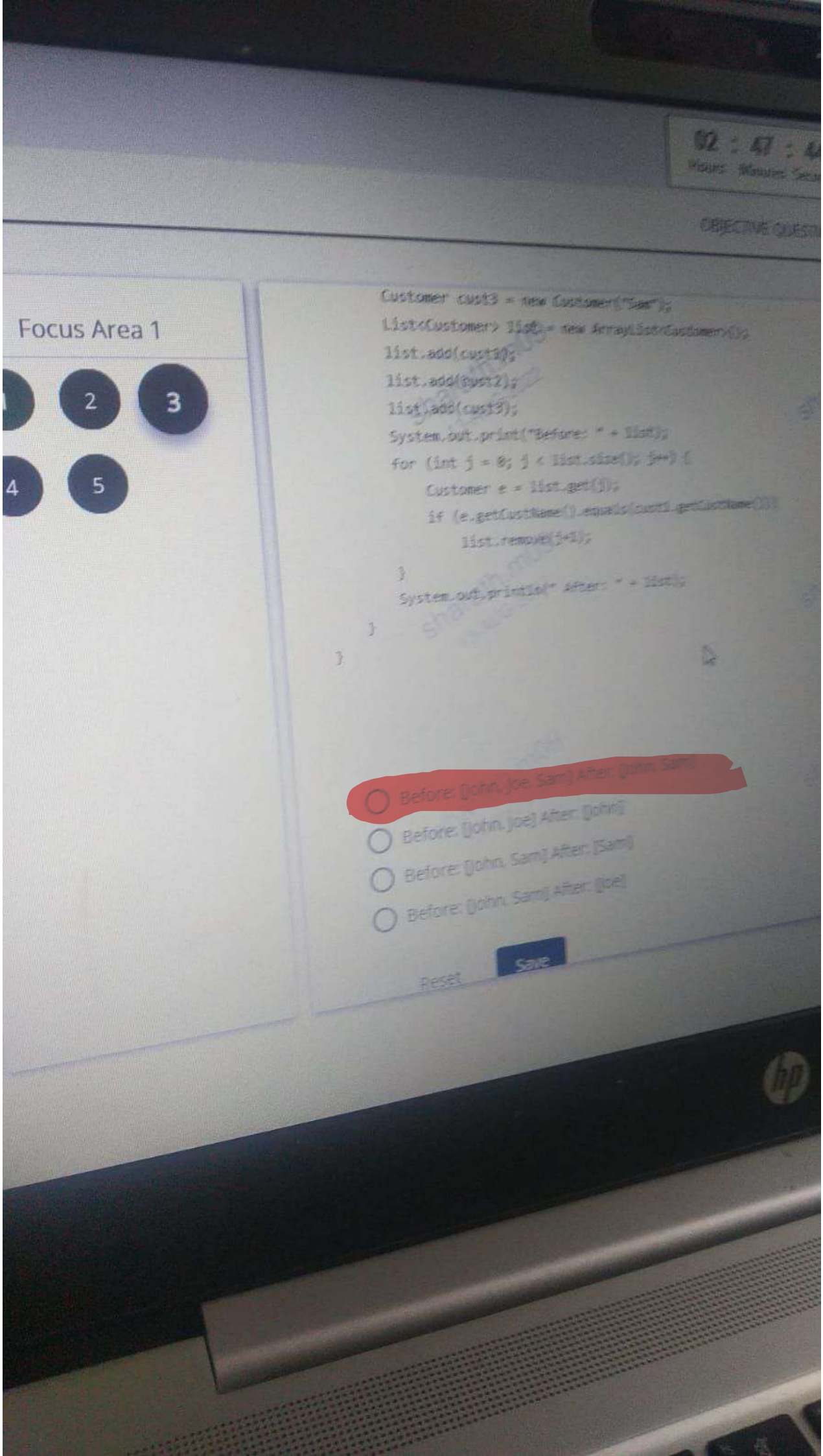
2 3

5

[2 Marks]

What is the output of the following code?

```
class Customer {
    private String custName;
    Customer(String fName) {
        this.custName = fName;
    }
    public String getCustName() {
        return custName;
    }
    public String toString() {
        return custName;
    }
}

public class Tester1 {
    public static void main(String[] args) {
        Customer cust1 = new Customer("John");
        Customer cust2 = new Customer("Joe");
        Customer cust3 = new Customer("Sam");
        List<Customer> list = new ArrayList<Customer>();
        list.add(cust1);
        list.add(cust2);
        list.add(cust3);
        System.out.print("Before: " + list);
        for (int j = 0; j < list.size(); j++) {
```


Focus Area 1
2
3
4
5
Customer cust3 = new Customer("Sam");
List<Customer> list = new ArrayList<Customer>();
list.add(cust1);
list.add(cust2);
list.add(cust3);
System.out.print("Before: " + list);
for (int j = 0; j < list.size(); j++) {
Customer e = list.get(j);
if (e.getCustName().equals(cust1.getCustName()))
list.remove(j+1);
}
System.out.println(" after: " + list);
}
}
Before: [John, Joe, Sam] After: [John, Sam]
Before: [John, Joe] After: [John]
Before: [John, Sam] After: [Sam]
Before: [John, Sam] After: [Joe]
Reset
Save

[2 Marks]

What will be the output of the below code? Assume **Customer class** exists as below

```
class Customer{
   int customerId;
   String customerName;
   Customer(int cId,String name){
       this.customerId=cId;
       this.customerName=name;
   }
   //Getters and setters
}
public class Tester2 {
    public static void main(String a[]) {
        List<Customer> list = new ArrayList<Customer>();
        Customer c1 = new Customer(9001, "Patrick");
        Customer c2 = new Customer(9002, "Mary");
        Customer c3 = new Customer(9003, "Anupam");
        list.add(c1);
        list.add(c2);
        list.add(c3);
        Stream<Customer> stream = list.stream().filter(cust -> cust.getCustomerName().length() >= 5);
        stream.forEach((customer) -> System.out.print(customer.getCustomerId() + " "));
    }
}
```

```
}
public class Tester2 {
    public static void main(String a[]) {
        List<Customer> list = new ArrayList<Customer>();
        Customer c1 = new Customer(9001, "Patrick");
        Customer c2 = new Customer(9002, "Mary");
        Customer c3 = new Customer(9003, "Anupam");
        list.add(c1);
        list.add(c2);
        list.add(c3);
        Stream<Customer> stream = list.stream().filter(cust -> cust.ge
        stream.forEach((customer) -> System.out.print(customer.getCust
    }
}
```


- ○ 9001
- ○ 9001 9003
- ○ 9001 9002 9003
- ○ 9002 9003

Reset Save

[1 Mark]

Assuming the below code executes in the main method. What will be the ou

```
LocalDate localDate = LocalDate.of(2020, 8, 12);
localDate = localDate.plusWeeks(4).minusMonths(1).plusYears(2);
System.out.println(localDate);
```

- ○ 2021-09-09
- ○ 2022-08-09
- ○ 2022-09-09
- ○ 2022-08-02

Reset Save

[2 Marks]

What will be the output when following code is executed?

```
public static void main(String args[]) {
        List<String> list = new LinkedList<String>();
        list.add("Earth");
        list.add("Jupiter");
        list.add("Mars");
        list.set(1, "Saturn");
        list.add("Earth");

        HashSet<String> set = new HashSet<String>();
        set.addAll(list);

        for (String element : set) {
                System.out.print(element+" ");
        }
}
```

Assumption:

1. All necessary configuration and imports are done

Note: The elements might appear in un-ordered fashion

```
Transaction transaction4 = new Transaction(1004, 1925.00, LocalDate.of(2021, 8, 15));
Transaction transaction5 = new Transaction(1005, 1206.00, LocalDate.of(2021, 8, 15));
transactions.add(transaction1);
transactions.add(transaction2);
transactions.add(transaction3);
transactions.add(transaction4);
transactions.add(transaction5);
Stream<Double> stream1 = transactions.stream().map(t -> t.getAmount());
Stream<Double> stream2 = stream1.filter(t -> t > 4000);
Stream<Double> stream3 = stream2.sorted();
List<Double> amountList = stream2.collect(Collectors.toList());
System.out.println(amountList);
    }
}
```

Assumption:

1. All necessary imports are done

- [5000.0, 56001.4]
- [28000.7]
- [2500.0, 28000.7]
- IllegalStateException exception will be thrown

Reset Save

What will be the output of the below code snippet?

```
public class DateTester {
        public static void main(String[] args) {
                DayOfWeek dayOfWeek = DayOfWeek.of(7);
                System.out.println(dayOfWeek);
        }
}
```

Assumption:

1. All necessary imports are done

- SATURDAY
- SUNDAY
- DateTimeException exception is thrown
- IndexOutOfBoundsException exception is thrown

Reset Save

What will be the output of the below program?

```
public class Tester{
   public static void main(String[] args) {
     Map <String,String> map1 = new LinkedHashMap<>();
     map1.put("1001", "Chandler");
     map1.put("1002", "Annie");
     map1.put("1010", "Joey");
     map1.remove(1010);
     map1.remove("1001");
     map1.remove(2);
      System.out.println(map1);
   }
}
```


Note: Assume the required imports are added.

- {1002=Annie}
- {1002=Annie, 1010=Joey}
- Error: ArrayIndexOutOfBounds Exception
- {1001=Chandler, 1002=Annie}

Reset Save

02 : 59 : 00
Hours Minutes Seconds

OBJECTIVE QUESTIONS

Focus Area 1

2 3

5

```
}

public class Tester {
    public static void main(String a[]) {
        List<Trainee> list = new ArrayList<Trainee>();
        Trainee c1 = new Trainee(110, "Patrick");
        Trainee c2 = new Trainee(112, "Chandler");
        Trainee c3 = new Trainee(113, "Darbie");
        Trainee c4 = new Trainee(115, "Kelly");
        list.add(c1);
        list.add(c2);
        list.add(c3);
        list.add(c4);
        if (list.contains(c3)) {
            list.set(1, list.get(3));
        }
        Stream<Trainee> stream = list.stream().filter(trainee -> trainee.getTraineeName().le
        stream.forEach((trainee) -> System.out.print(trainee.getTraineeId() + " "));
    }
}
```


110 113

110 112 113

02 : 59 : 13
Hours Minutes Seconds

OBJECTIVE QUESTIONS

Focus Area 1

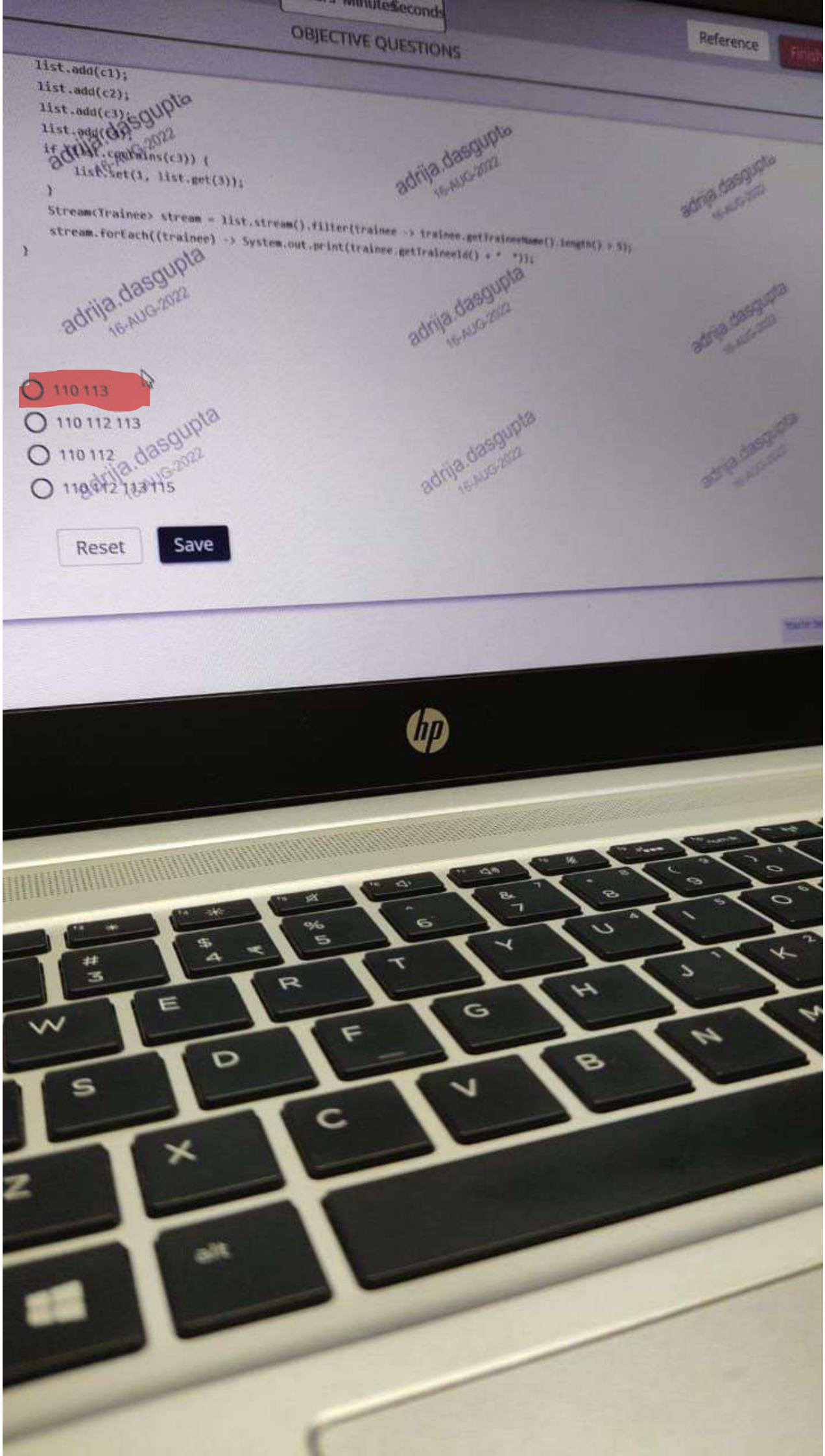
2 3 5

Question 5

[2 Marks]

What will be the output of the below code? Assume **Trainee class** exists as below

```
class Trainee {
    int traineeId;
    String traineeName;
    Trainee(int cId, String name) {
        this.traineeId = cId;
        this.traineeName = name;
    }
    //getters and setters
}

public class Tester {
    public static void main(String a[]) {
        List<Trainee> list = new ArrayList<Trainee>();
        Trainee c1 = new Trainee(110, "Patrick");
        Trainee c2 = new Trainee(112, "Chandler");
        Trainee c3 = new Trainee(113, "Barbie");
        Trainee c4 = new Trainee(115, "Kelly");
        list.add(c1);
        list.add(c2);
        list.add(c3);
```


OBJECTIVE QUESTIONS
Reference
list.add(c1);
list.add(c2);
list.add(c3);
list.add(c4);
if (list.contains(c3)) {
list.set(1, list.get(3));
}
Stream<Trainee> stream = list.stream().filter(trainee -> trainee.getTraineeName().length() > 5);
stream.forEach((trainee) -> System.out.print(trainee.getTraineeId() + " "));
}
110 113
110 112 113
110 112
110 112 113 115
Reset
Save
adrija.dasgupta
16-AUG-2022

```
c class Tester{
public static void main(String[] args) {
    Employee emp1 = new Employee("John");
    Employee emp2 = new Employee("Joe");
    Employee emp3 = new Employee("Sam");
    List<Employee> list = new ArrayList<Employee>();
    list.add(emp1);
    list.add(emp2);
    list.add(emp3);
    System.out.print("Before: " + list);
    for (int j = 0; j < list.size(); j++) {
        Employee e = list.get(j);
        if (e.getFirstName().equals(emp1.getFirstName()
            list.remove(j);

    }
    System.out.println(" After: " + list);

  }

}
```

Before: [John, Joe, Sam] After: [Joe, Sam]

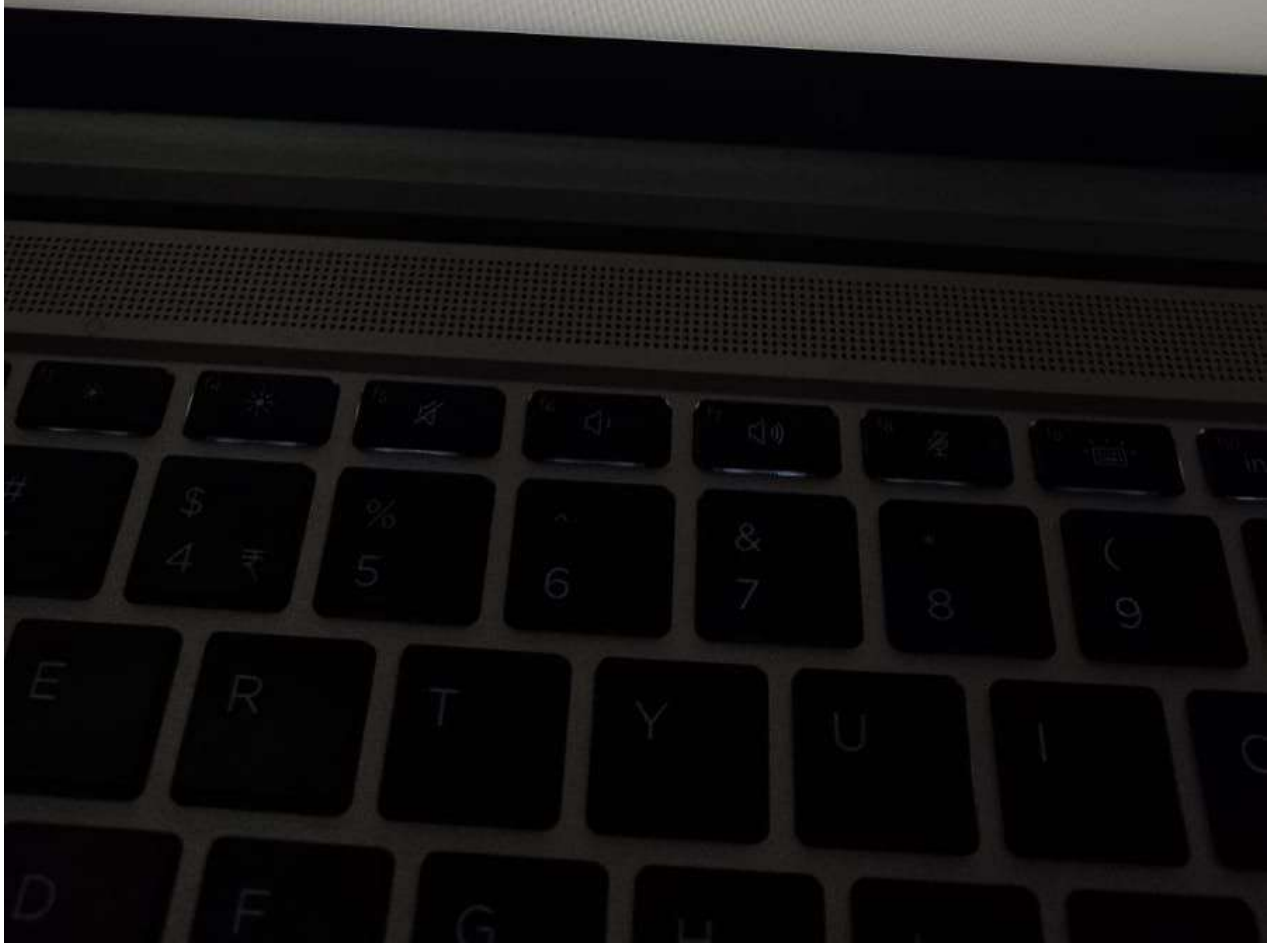
```
public static void main(String[] args) {
        Map<String, String> nameMap = new HashMap<String, String>();

        nameMap.put("A", "Andre");
        nameMap.put("B", "Bob");
        nameMap.put(null, null);
        nameMap.put("C", "Catlyn");
        nameMap.put(new String("A"), "Avan");
        System.out.println(nameMap);
}
```

Assumption:

1. All necessary configuration and imports are done

- {null=null, A=Avan, B=Bob, C=Catlyn}
- {A=Avan, B=Bob, C=Catlyn}
- The code will throw run time exception as the same key cannot be used for 2 va
- The code will throw NullPointerException as the HashMap cannot have null as ke

Reset Save



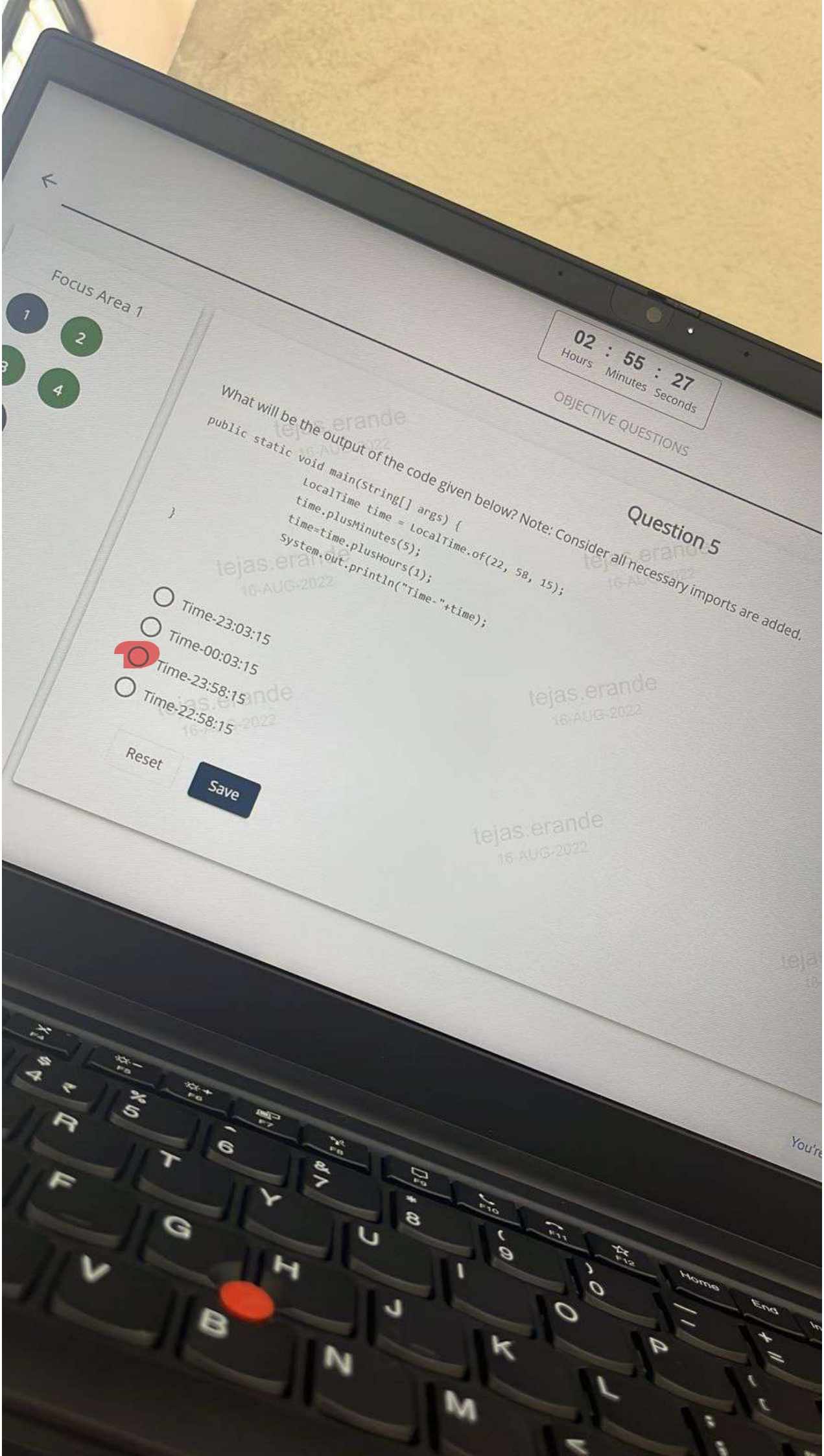
```
public static void main(String[] args) {
        LocalTime newTimeObj = LocalTime.of(9, 45, 59);

        int hour = newTimeObj.getHour();
        int second = newTimeObj.getSecond();
        int minute = newTimeObj.getMinute();
        System.out.println(hour+":"+second+":"+minute);
    }
```

Assumption:

1. All necessary configuration and imports are done

- ○ 9:45:59
- ○ 9:59:45
- ○ 09:45:59
- ○ 09:59:45

Reset Save


Focus Area 1
1
2
4
02 : 55 : 27
Hours Minutes Seconds
OBJECTIVE QUESTIONS
Question 5
What will be the output of the code given below? Note: Consider all necessary imports are added.
public static void main(String[] args) {
LocalTime time = LocalTime.of(22, 58, 15);
time.plusMinutes(5);
time=time.plusHours(1);
System.out.println("Time-"+time);
}
Time-23:03:15
Time-00:03:15
Time-23:58:15
Time-22:58:15
Reset
Save
tejas.erande
16-AUG-2022

4

Question 1

What will be the output of the following code snippet?

```
public class Tester {
    public static void main(String[] args) {
        Map map = new TreeMap();
        map.put("Spring", 10);
        map.put("Java", 20);
        map.put("spring", 20);
        map.put("Java", 50);
        System.out.println(map);
    }
}
```

Note: Consider all necessary imports are added.

- {Java=20, Spring=10, spring=20}
- {Java=50, Spring=10, spring=20}
- {Spring=10, Java=50, spring=20}
- {Java=50, spring=20}

java =50, spring 20

[2 Marks]

Consider the below code.

```
public class CollectionsTest {
        public static void main(String[] args) {
                List<String> numList = new ArrayList<>();
                int listSize = numList.size();
                listSize += 10;
                numList.add("Java");
                for (int i = 1; i < listSize; i++) {
                        numList.add(numList.get(i - 1) + "-Java");
                        numList.add(++i, "Java");
                }
                numList.remove(2);
                System.out.println("Element at 3rd position: " + numList.get(2));
                System.out.println("Size: " + numList.size());
        }
}
```

What will be the output of above code snippet? Note: Consider all necessary imports are added.

- Element at 3rd position: Java-Java
  Size: 10
- Element at 3rd position: Java-Java
  Size: 9
- Element at 3rd position: Java
  Size: 9
- Element at 3rd position: Java-Java-Java

Assume that all necessary imports have been added.

```
public class Tester {
  public static void main(String[] args) {
     Map<Integer, String> map = new TreeMap<>();
     map.put(101, "King");
     map.put(101, "Queen");
     map.put(103, "Ace");
     map.put(102, "Jack");
     map.put(104, "Queen");
     System.out.println(map);
  }
}
```

What will be the output of the above code?

- {101=Queen, 102=Jack, 103=Ace, 104=Queen}
- {101=King, 102=Jack, 103=Ace, 104=Queen}
- {103=Ace, 102=Jack, 101=King, 104=Queen}
- {103=Ace, 102=Jack, 101=Queen, 104=Queen}