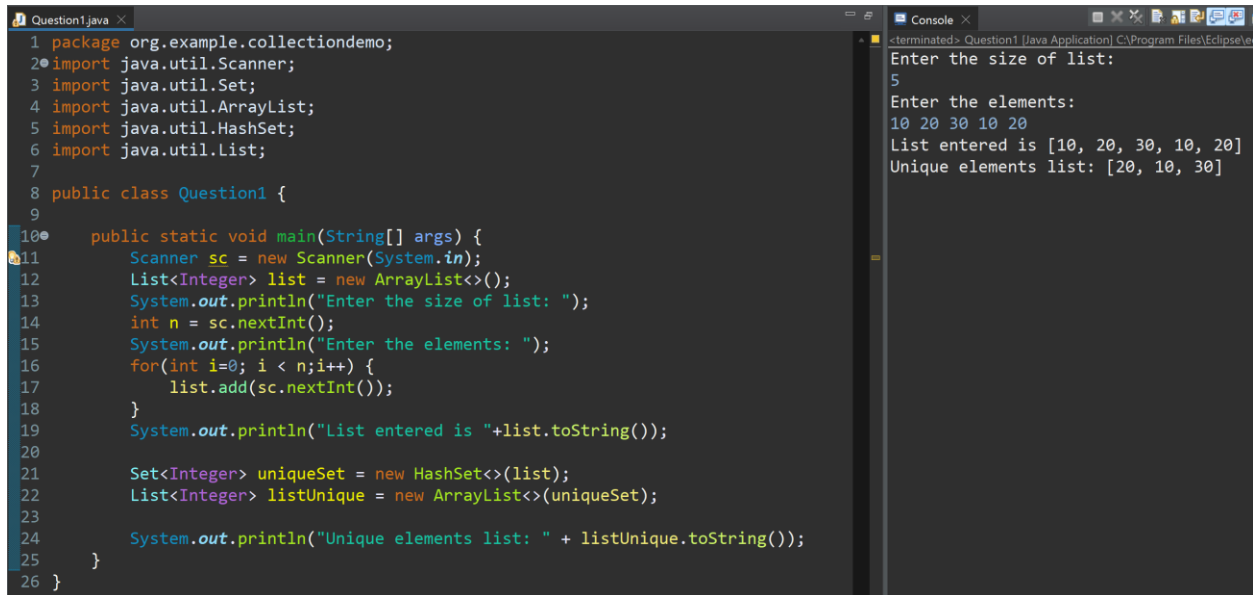


PG-DAC AUGUST 24 BATCH

1) Write a Java program that takes a list of integers as input and returns a list of duplicate integers.

The screenshot shows an IDE with two panels. The left panel displays a Java file named 'Question1.java' with the following code:

```
1 package org.example.collectiondemo;
2 import java.util.Scanner;
3 import java.util.Set;
4 import java.util.ArrayList;
5 import java.util.HashSet;
6 import java.util.List;
7
8 public class Question1 {
9
10     public static void main(String[] args) {
11         Scanner sc = new Scanner(System.in);
12         List<Integer> list = new ArrayList<>();
13         System.out.println("Enter the size of list: ");
14         int n = sc.nextInt();
15         System.out.println("Enter the elements: ");
16         for(int i=0; i < n; i++) {
17             list.add(sc.nextInt());
18         }
19         System.out.println("List entered is " + list.toString());
20
21         Set<Integer> uniqueSet = new HashSet<>(list);
22         List<Integer> listUnique = new ArrayList<>(uniqueSet);
23
24         System.out.println("Unique elements list: " + listUnique.toString());
25     }
26 }
```

The right panel shows the console output:

```
<terminated> Question1 [Java Application] C:\Program Files\Eclipse\
Enter the size of list:
5
Enter the elements:
10 20 30 10 20
List entered is [10, 20, 30, 10, 20]
Unique elements list: [20, 10, 30]
```

Code:

```
package org.example.collectiondemo;
import java.util.Scanner;
import java.util.Set;
import java.util.ArrayList;
import java.util.HashSet;
import java.util.List;
public class Question1 {
    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);
        List<Integer> list = new ArrayList<>();
        System.out.println("Enter the size of list: ");
        int n = sc.nextInt();
        System.out.println("Enter the elements: ");
        for(int i=0; i < n; i++) {
            list.add(sc.nextInt());
        }
        System.out.println("List entered is " + list.toString());

        Set<Integer> uniqueSet = new HashSet<>(list);
        List<Integer> listUnique = new ArrayList<>(uniqueSet);

        System.out.println("Unique elements list: " + listUnique.toString());
    }
}
```

2) Create a Person class with attributes name and age. Write a Java program that sorts a list of Person objects first by age and then by name if the ages are equal.

```
1 package org.example.collectiondemo;
2
3 import java.util.ArrayList;
4 import java.util.Collections;
5 import java.util.Comparator;
6 import java.util.List;
7
8 class Person {
9     private String name;
10    private int age;
11
12    public Person(String name, int age) {
13        this.name = name;
14        this.age = age;
15    }
16
17    public String getName() {
18        return name;
19    }
20
21    public int getAge() {
22        return age;
23    }
24
25    @Override
26    public String toString() {
27        return "Person{name='" + name + "', age=" + age + "}";
28    }
29 }
30
31 public class Question2 {
32    public static void main(String[] args) {
33
34        List<Person> people = new ArrayList<>();
35        people.add(new Person("Ana", 30));
36        people.add(new Person("Minal", 25));
37        people.add(new Person("Chia", 30));
38        people.add(new Person("Devi", 20));
39        people.add(new Person("Archie", 25));
40
41        Collections.sort(people, new Comparator<Person>() {
42            @Override
43            public int compare(Person p1, Person p2) {
44                // Compare by age
45                int ageComparison = Integer.compare(p1.getAge(), p2.getAge());
46                if (ageComparison == 0) {
47                    // If ages are equal, compare by name
48                    return p1.getName().compareTo(p2.getName());
49                }
50                return ageComparison;
51            }
52        });
53
54        System.out.println("Sorted list of people:");
55        for (Person person : people) {
56            System.out.println(person);
57        }
58    }
59 }
```

Code:

```
package org.example.collectiondemo;
```

```
import java.util.ArrayList;
import java.util.Collections;
import java.util.Comparator;
import java.util.List;
```

```
class Person {
    private String name;
    private int age;
```

```
    public Person(String name, int age) {
        this.name = name;
```

```

        this.age = age;
    }

    public String getName() {
        return name;
    }

    public int getAge() {
        return age;
    }

    @Override
    public String toString() {
        return "Person{name='" + name + "', age=" + age + "'}";
    }
}

public class Question2 {
    public static void main(String[] args) {

        List<Person> people = new ArrayList<>();
        people.add(new Person("Ana", 30));
        people.add(new Person("Minal", 25));
        people.add(new Person("Chia", 30));
        people.add(new Person("Devi", 20));
        people.add(new Person("Archie", 25));

        Collections.sort(people, new Comparator<Person>() {
            @Override
            public int compare(Person p1, Person p2) {
                // Compare by age
                int ageComparison = Integer.compare(p1.getAge(), p2.getAge());
                if (ageComparison == 0) {
                    // If ages are equal, compare by name
                    return p1.getName().compareTo(p2.getName());
                }
                return ageComparison;
            }
        });

        System.out.println("Sorted list of people:");
        for (Person person : people) {
            System.out.println(person);
        }
    }
}

```

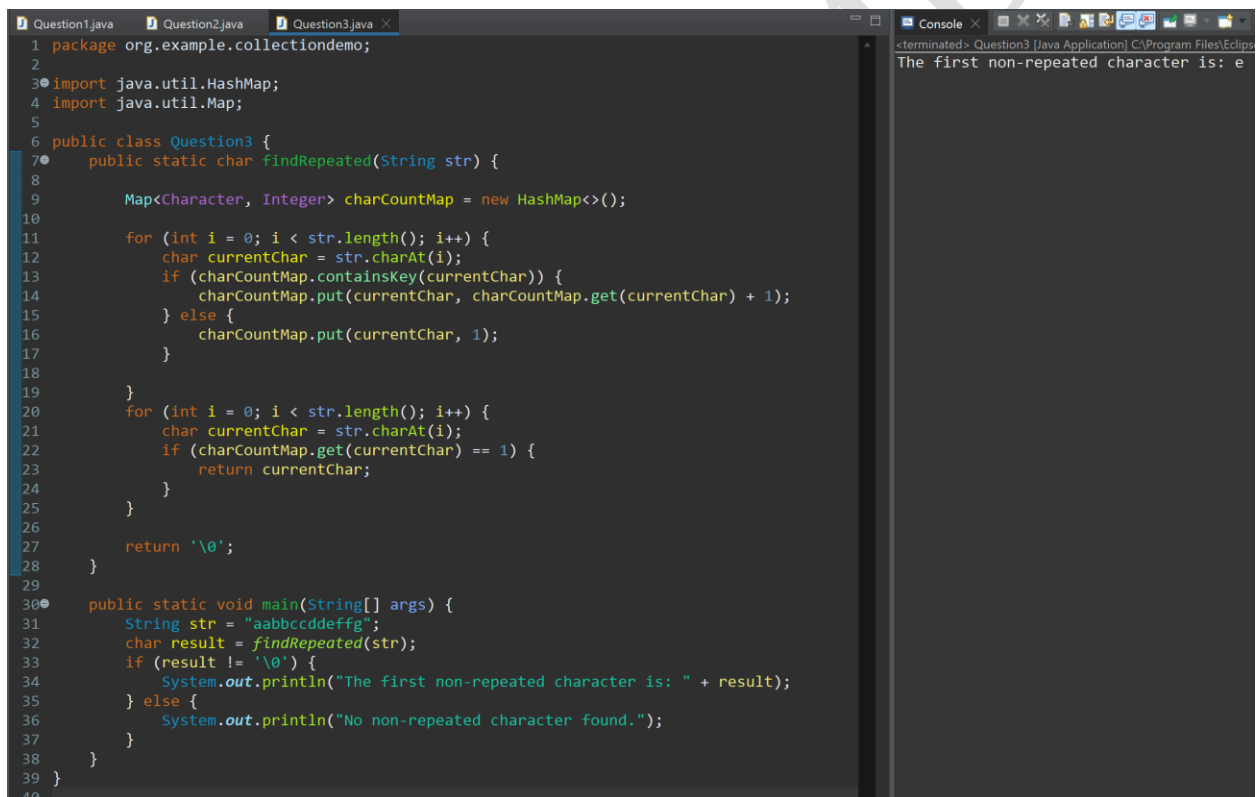
Output:

```
Sorted list of people:
Person{name='Devi', age=20}
Person{name='Archie', age=25}
Person{name='Minal', age=25}
Person{name='Ana', age=30}
Person{name='Chia', age=30}
```

3) Write a Java program to find the first non-repeated character in a string using a HashMap.

String input = "aabbccddeffg";

Expected output = 'e';



```
1 package org.example.collectiondemo;
2
3 import java.util.HashMap;
4 import java.util.Map;
5
6 public class Question3 {
7     public static char findRepeated(String str) {
8
9         Map<Character, Integer> charCountMap = new HashMap<>();
10
11         for (int i = 0; i < str.length(); i++) {
12             char currentChar = str.charAt(i);
13             if (charCountMap.containsKey(currentChar)) {
14                 charCountMap.put(currentChar, charCountMap.get(currentChar) + 1);
15             } else {
16                 charCountMap.put(currentChar, 1);
17             }
18         }
19
20         for (int i = 0; i < str.length(); i++) {
21             char currentChar = str.charAt(i);
22             if (charCountMap.get(currentChar) == 1) {
23                 return currentChar;
24             }
25         }
26
27         return '\0';
28     }
29
30     public static void main(String[] args) {
31         String str = "aabbccddeffg";
32         char result = findRepeated(str);
33         if (result != '\0') {
34             System.out.println("The first non-repeated character is: " + result);
35         } else {
36             System.out.println("No non-repeated character found.");
37         }
38     }
39 }
40
```

Console output: <terminated> Question3 [Java Application] C:\Program Files\Eclipse...
The first non-repeated character is: e

Code: package org.example.collectiondemo;

import java.util.HashMap;

import java.util.Map;

```
public class Question3 {
    public static char findRepeated(String str) {
```

```
        Map<Character, Integer> charCountMap = new HashMap<>();
```

```

    for (int i = 0; i < str.length(); i++) {
        char currentChar = str.charAt(i);
        if (charCountMap.containsKey(currentChar)) {
            charCountMap.put(currentChar, charCountMap.get(currentChar) + 1);
        } else {
            charCountMap.put(currentChar, 1);
        }
    }
    for (int i = 0; i < str.length(); i++) {
        char currentChar = str.charAt(i);
        if (charCountMap.get(currentChar) == 1) {
            return currentChar;
        }
    }

    return '\0';
}

public static void main(String[] args) {
    String str = "aabbccddeffg";
    char result = findRepeated(str);
    if (result != '\0') {
        System.out.println("The first non-repeated character is: " + result);
    } else {
        System.out.println("No non-repeated character found.");
    }
}
}

```

4) Write a Java program that merges two sorted lists of integers into a single sorted list.

Code:

```

package org.example.collectiondemo;
import java.util.ArrayList;
import java.util.List;

public class Question4 {

    public static List<Integer> mergeTwoSortedLists(List<Integer> list1, List<Integer> list2) {
        List<Integer> mergedList = new ArrayList<>();
        int i = 0, j = 0;

        // Merge both lists until one of them is finish
        while (i < list1.size() && j < list2.size()) {
            if (list1.get(i) < list2.get(j)) {
                mergedList.add(list1.get(i));
                i++;
            } else {
                mergedList.add(list2.get(j));
            }
        }
    }
}

```

```
        j++;
    }
}

// Add remaining elements from list1
while (i < list1.size()) {
    mergedList.add(list1.get(i));
    i++;
}

// Add remaining elements from list2
while (j < list2.size()) {
    mergedList.add(list2.get(j));
    j++;
}

return mergedList;
}

public static void main(String[] args) {

    List<Integer> list1 = new ArrayList<>();
    list1.add(1);
    list1.add(3);
    list1.add(5);

    List<Integer> list2 = new ArrayList<>();
    list2.add(2);
    list2.add(4);
    list2.add(6);
    List<Integer> mergedList = mergeTwoSortedLists(list1, list2);

    System.out.println("Merged Sorted List: " + mergedList);
}
}
```

```

1 package org.example.collectiondemo;
2 import java.util.ArrayList;
3 import java.util.List;
4
5 public class Question4 {
6
7     public static List<Integer> mergeTwoSortedLists(List<Integer> list1, List<Integer> list2) {
8         List<Integer> mergedList = new ArrayList<>();
9         int i = 0, j = 0;
10
11         // Merge both lists until one of them is finish
12         while (i < list1.size() && j < list2.size()) {
13             if (list1.get(i) < list2.get(j)) {
14                 mergedList.add(list1.get(i));
15                 i++;
16             } else {
17                 mergedList.add(list2.get(j));
18                 j++;
19             }
20         }
21
22         // Add remaining elements from list1
23         while (i < list1.size()) {
24             mergedList.add(list1.get(i));
25             i++;
26         }
27
28         // Add remaining elements from list2
29         while (j < list2.size()) {
30             mergedList.add(list2.get(j));
31             j++;
32         }
33
34         return mergedList;
35     }
36
37     public static void main(String[] args) {
38
39         List<Integer> list1 = new ArrayList<>();
40         list1.add(1);
41         list1.add(3);
42         list1.add(5);
43
44         List<Integer> list2 = new ArrayList<>();
45         list2.add(2);
46         list2.add(4);
47         list2.add(6);
48
49
50         List<Integer> mergedList = mergeTwoSortedLists(list1, list2);
51
52         System.out.println("Merged Sorted List: " + mergedList);
53     }
54 }

```

Console

<terminated> Question4 [Java Application] C:\Program Files\Eclipse\eclipse\plug
Merged Sorted List: [1, 2, 3, 4, 5, 6]