

LAB NO 2

DATA STRUCTURES AND ALGORITHMS

OBJECTIVE: To implement ArrayList and Vector.

TASK NO 1:

Write a program that initializes Vector with 10 integers in it. Display all the integers and sum of these integers.

INPUT:

```
package shaheer.javaaid;
import java.util.*;
public class ShaheerJavaid {
    public static void main(String[] args) {

        Vector<Integer> vector = new Vector<>();
        int sum = 0;

        for (int i = 1; i <= 10; i++) {
            vector.add(i);
            sum += i;
        }

        System.out.println("Vector Elements: " + vector);
        System.out.println("Sum of Elements: " + sum);
    }
}
```

OUTPUT:

Output - Shaheer Javaid (run) ×



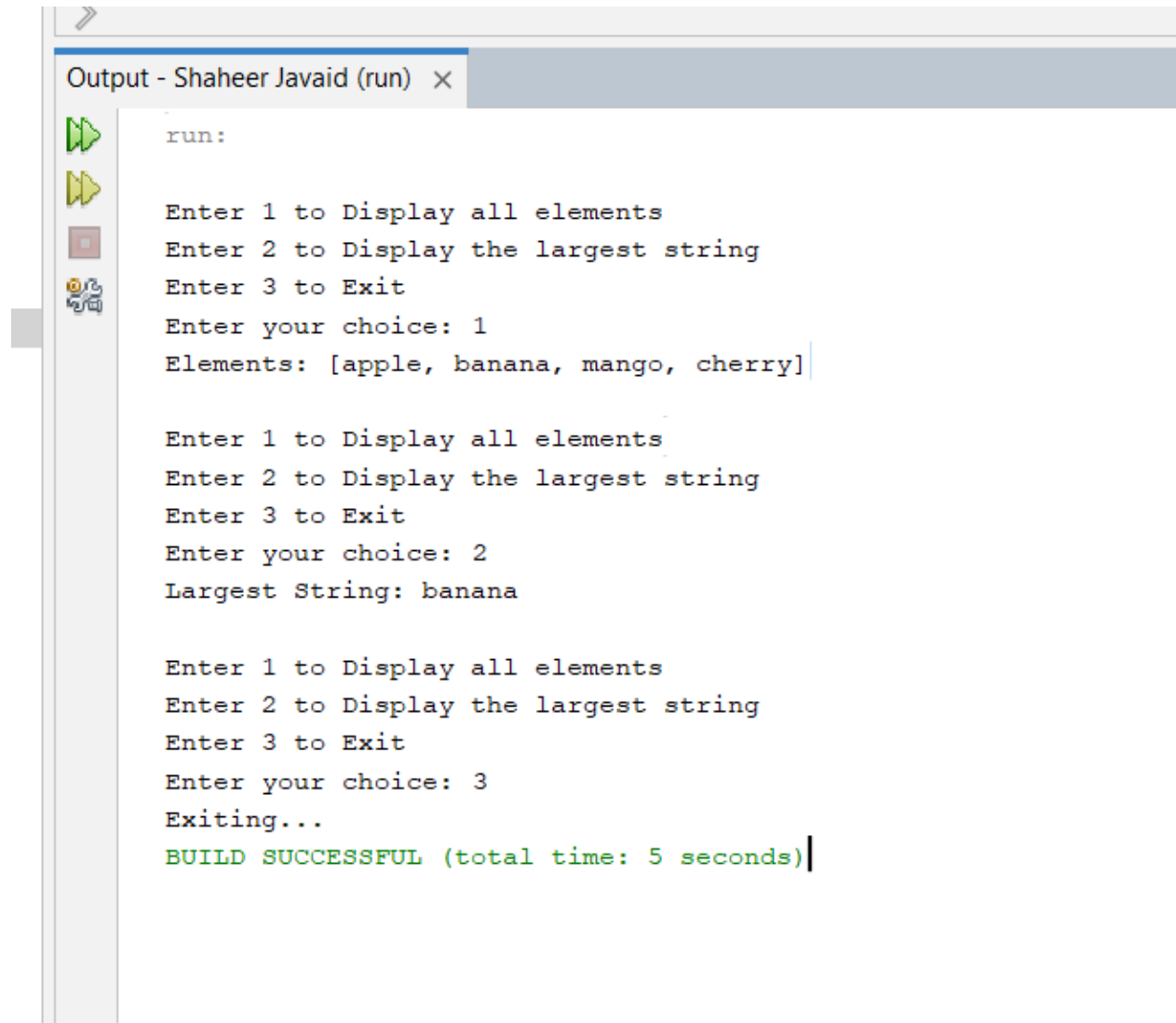
```
run:
Vector Elements: [1, 2, 3, 4, 5, 6, 7, 8, 9, 10]
Sum of Elements: 55
BUILD SUCCESSFUL (total time: 0 seconds)
|
```

TASK NO 2: Create a ArrayList of string. Write a menu driven program which: a. Displays all the elements b. Displays the largest String

INPUT:

```
1  /
2  package shaheer.javaaid;
3  import java.util.*;
4  public class ShaheerJavaid {
5      public static void main(String[] args) {
6
7          ArrayList<String> strings = new ArrayList<>(Arrays.asList("apple", "banana", "mango", "cherry"));
8          Scanner scanner = new Scanner(System.in);
9          int choice;
10
11          do {
12              System.out.println("\nEnter 1 to Display all elements");
13              System.out.println("Enter 2 to Display the largest string");
14              System.out.println("Enter 3 to Exit");
15              System.out.print("Enter your choice: ");
16              choice = scanner.nextInt();
17
18              switch (choice) {
19                  case 1:
20                      System.out.println("Elements: " + strings);
21                      break;
22                  case 2:
23                      String largest = Collections.max(strings, Comparator.comparing(String::length));
24                      System.out.println("Largest String: " + largest);
25                      break;
26                  case 3:
27                      System.out.println("Exiting...");
28                      break;
29                  default:
30                      System.out.println("Invalid choice. Try again.");
31              }
32          } while (choice != 3);
33      }
34  }
```

OUTPUT:



```
run:
Enter 1 to Display all elements
Enter 2 to Display the largest string
Enter 3 to Exit
Enter your choice: 1
Elements: [apple, banana, mango, cherry]

Enter 1 to Display all elements
Enter 2 to Display the largest string
Enter 3 to Exit
Enter your choice: 2
Largest String: banana

Enter 1 to Display all elements
Enter 2 to Display the largest string
Enter 3 to Exit
Enter your choice: 3
Exiting...
BUILD SUCCESSFUL (total time: 5 seconds)
```

TASK NO 3:

Create a ArrayList storing Employee details including Emp_id, Emp_Name, Emp_gender, Year_of_Joining (you can also add more attributes including these). Then sort the employees according to their joining year using Comparator and Comparable interfaces.

INPUT:

```
1  /*
2  package shaheer.javaaid;
3
4  import java.util.*;
5
6  class Employee {
7      int empId;
8      String empName;
9      String empGender;
10     int yearOfJoining;
11
12     Employee(int empId, String empName, String empGender, int yearOfJoining) {
13         this.empId = empId;
14         this.empName = empName;
15         this.empGender = empGender;
16         this.yearOfJoining = yearOfJoining;
17     }
18
19     @Override
20     public String toString() {
21         return "ID: " + empId + ", Name: " + empName + ", Gender: " + empGender + ", Year: " + yearOfJoining;
22     }
23 }
24
25 public class ShaheerJavaid {
26     public static void main(String[] args) {
27         Scanner scanner = new Scanner(System.in);
28         ArrayList<Employee> employees = new ArrayList<>();
29
30
31         System.out.print("Enter the number of employees: ");
32         int n = scanner.nextInt();
33
34         for (int i = 0; i < n; i++) {
35             System.out.println("\nEnter details for employee " + (i + 1) + ":");
36             System.out.print("Enter Employee ID: ");
37             int empId = scanner.nextInt();
38             scanner.nextLine();
39
40             System.out.print("Enter Employee Name: ");
41             String empName = scanner.nextLine();
42
43             System.out.print("Enter Employee Gender: ");
44             String empGender = scanner.nextLine();
45
46             System.out.print("Enter Year of Joining: ");
47             int yearOfJoining = scanner.nextInt();
48
49             employees.add(new Employee(empId, empName, empGender, yearOfJoining));
50         }
51
52         employees.sort(Comparator.comparingInt(e -> e.yearOfJoining));ss
53         System.out.println("\nEmployees sorted by year of joining:");
54         for (Employee emp : employees) {
55             System.out.println(emp);
56         }
57     }
58 }
```

OUTPUT:The screenshot shows a Java IDE's output window. The title bar reads "Output - Shaheer Javaid (run)". On the left, there are icons for running (green play button), stepping through (yellow play button), stopping (red square), and debugging (bug icon). The output text is as follows:

```
run:
Enter the number of employees: 2
Enter details for employee 1:
Enter Employee ID: 2364
Enter Employee Name: Moosa
Enter Employee Gender: Male
Enter Year of Joining: 2024

Enter details for employee 2:
Enter Employee ID: 1453
Enter Employee Name: Shaheer
Enter Employee Gender: Male
Enter Year of Joining: 2023

Employees sorted by year of joining:
ID: 1453, Name: Shaheer, Gender: Male, Year: 2023
ID: 2364, Name: Moosa, Gender: Male, Year: 2024
BUILD SUCCESSFUL (total time: 52 seconds)
```

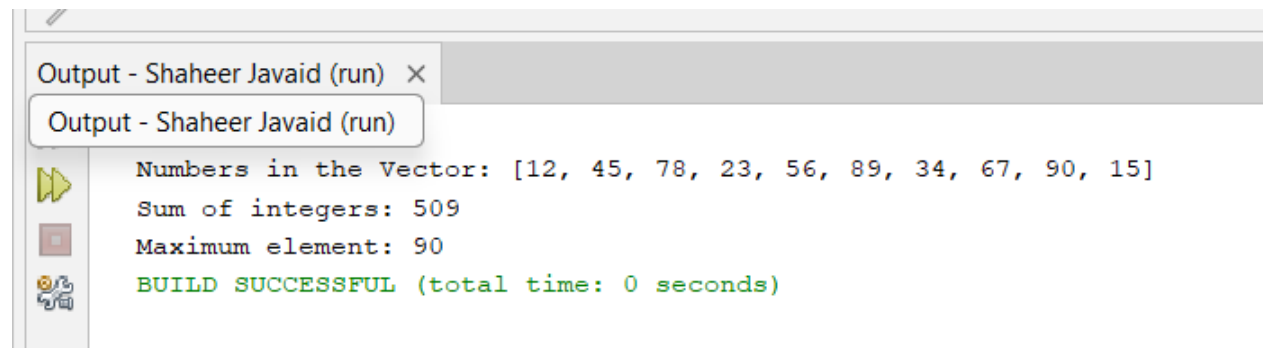
TASK NO 4:

Write a program that initializes Vector with 10 integers in it.

- Display all the integers
- Sum of these integers.
- Find Maximum Element in Vector

INPUT:

```
1 //  
2 package shaheer.javaaid;  
3 import java.util.*;  
4  
5 public class ShaheerJavaid {  
6     public static void main(String[] args) {  
7         Vector<Integer> numbers = new Vector<>(Arrays.asList(12, 45, 78, 23, 56, 89, 34, 67, 90, 15));  
8  
9         System.out.println("Numbers in the Vector: " + numbers);  
10  
11         int sum = 0;  
12         for (int num : numbers) {  
13             sum += num;  
14         }  
15         System.out.println("Sum of integers: " + sum);  
16  
17         int max = Collections.max(numbers);  
18         System.out.println("Maximum element: " + max);  
19     }  
20 }
```

OUTPUT:

The screenshot shows an IDE output window titled "Output - Shaheer Javaid (run)". The output text is as follows:

```
Numbers in the Vector: [12, 45, 78, 23, 56, 89, 34, 67, 90, 15]  
Sum of integers: 509  
Maximum element: 90  
BUILD SUCCESSFUL (total time: 0 seconds)
```

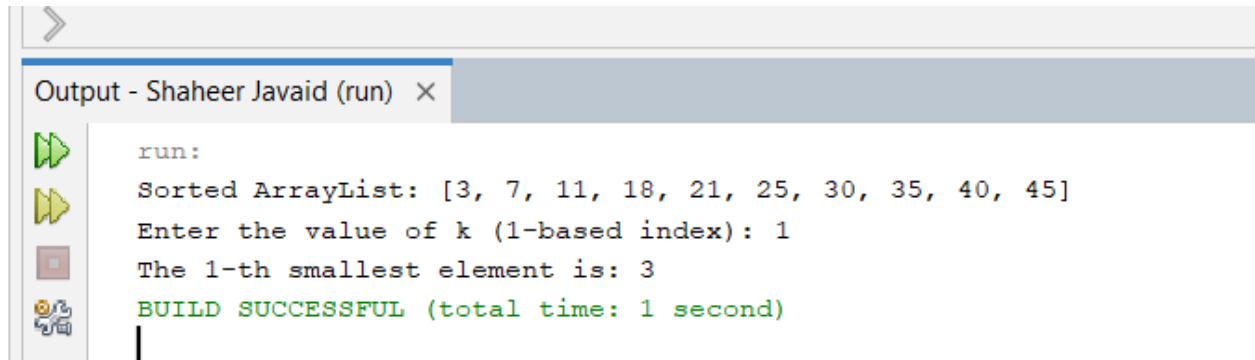
:

TASK NO 5

Find the k-th smallest element in a sorted ArrayList

INPUT:

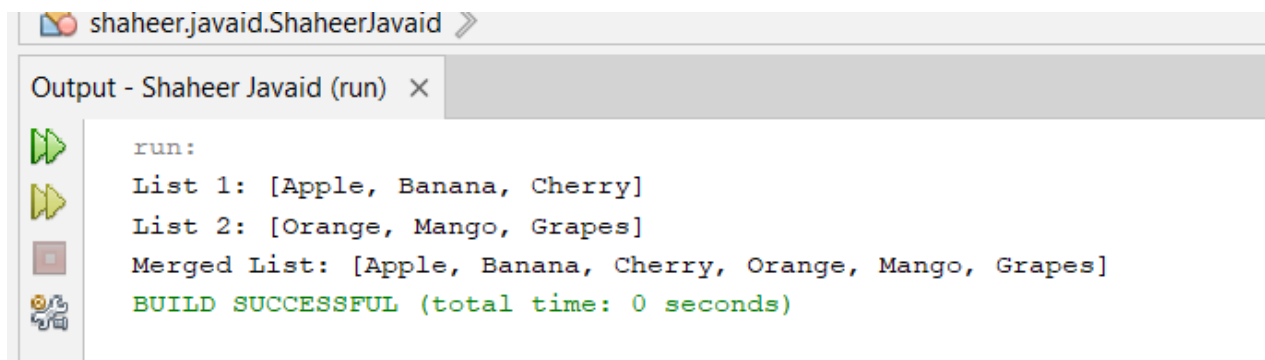
```
public class ShaheerJavaid {  
    public static void main(String[] args) {  
        ArrayList<Integer> numbers = new ArrayList<>(Arrays.asList(3, 7, 11, 18, 21, 25, 30, 35, 40, 45));  
  
        System.out.println("Sorted ArrayList: " + numbers);  
  
        Scanner scanner = new Scanner(System.in);  
        System.out.print("Enter the value of k (1-based index): ");  
        int k = scanner.nextInt();  
  
        if (k > 0 && k <= numbers.size()) {  
            int kthSmallest = numbers.get(k - 1);  
            System.out.println("The " + k + "-th smallest element is: " + kthSmallest);  
        } else {  
            System.out.println("Invalid value of k. Please enter a value between 1 and " + numbers.size());  
        }  
    }  
}
```

OUTPUT:**TASK NO 6:**

Write a program to merge two ArrayLists into one.

INPUT :

```
/*  
package shaheer.javaaid;  
import java.util.*;  
  
public class ShaheerJavaid {  
    public static void main(String[] args) {  
        ArrayList<String> list1 = new ArrayList<>(Arrays.asList("Apple", "Banana", "Cherry"));  
        System.out.println("List 1: " + list1);  
  
        ArrayList<String> list2 = new ArrayList<>(Arrays.asList("Orange", "Mango", "Grapes"));  
        System.out.println("List 2: " + list2);  
  
        ArrayList<String> mergedList = new ArrayList<>(list1);  
        mergedList.addAll(list2);  
  
        System.out.println("Merged List: " + mergedList);  
    }  
}
```

OUTPUT:**HOME TASKS****TASK NO 1:**

Create a Vector storing integer objects as an input

- a. Sort the vector
- b. Display largest number
- c. Display smallest number

INPUT:

:

```
package shaheer.javaaid;
import java.util.*;

public class ShaheerJavaid {
    public static void main(String[] args) {
        Scanner scanner = new Scanner(System.in);
        Vector<Integer> numbers = new Vector<>();

        System.out.println("Enter 10 integers:");
        for (int i = 0; i < 10; i++) {
            System.out.print("Enter number " + (i + 1) + ": ");
            numbers.add(scanner.nextInt());
        }

        Collections.sort(numbers);

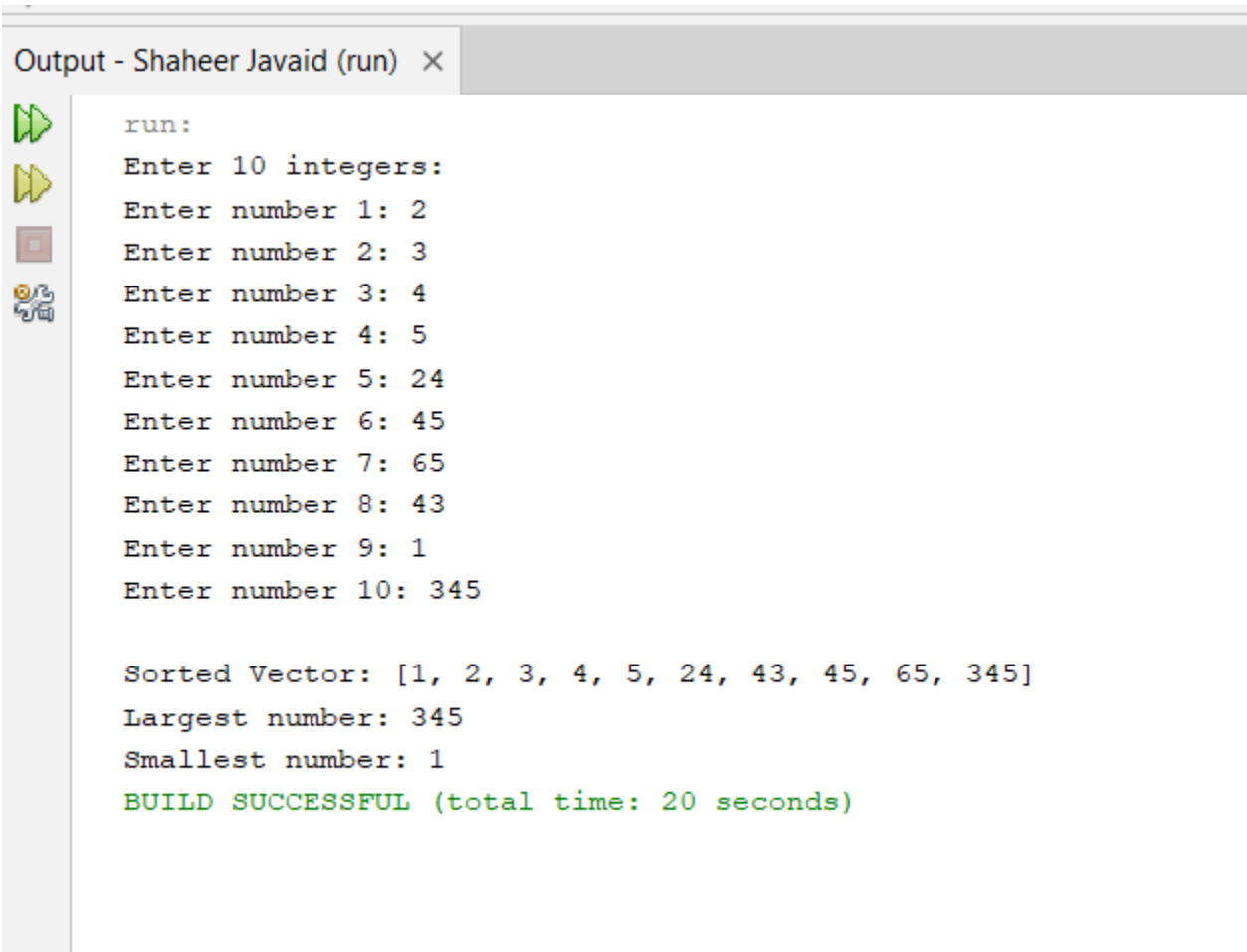
        System.out.println("\nSorted Vector: " + numbers);

        int largest = Collections.max(numbers);
        System.out.println("Largest number: " + largest);

        int smallest = Collections.min(numbers);
        System.out.println("Smallest number: " + smallest);

        scanner.close();
    }
}
```

OUTPUT:



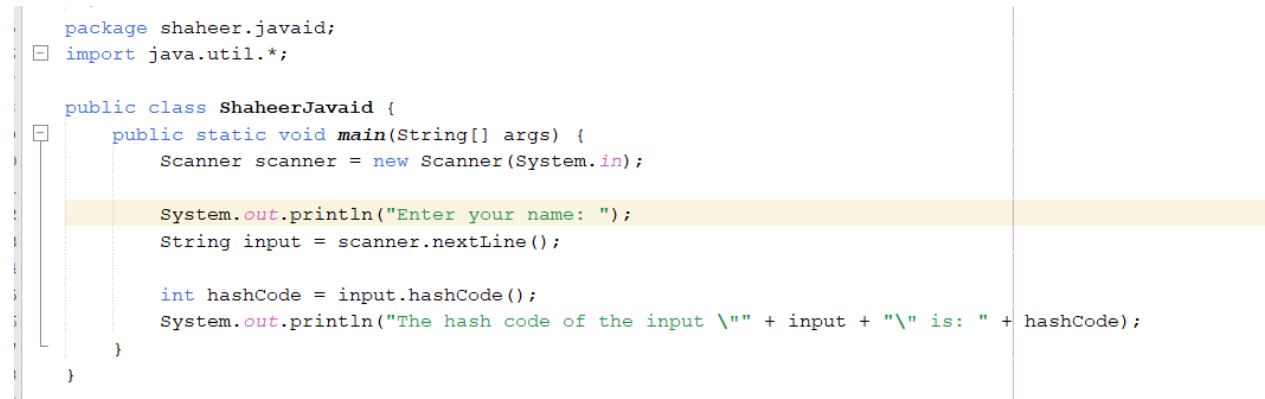
```
run:
Enter 10 integers:
Enter number 1: 2
Enter number 2: 3
Enter number 3: 4
Enter number 4: 5
Enter number 5: 24
Enter number 6: 45
Enter number 7: 65
Enter number 8: 43
Enter number 9: 1
Enter number 10: 345

Sorted Vector: [1, 2, 3, 4, 5, 24, 43, 45, 65, 345]
Largest number: 345
Smallest number: 1
BUILD SUCCESSFUL (total time: 20 seconds)
```

TASK NO 2

Write a java program which takes user input and gives hashcode value of those inputs using hashCode () method.

INPUT:



```
package shaheer.javaaid;
import java.util.*;

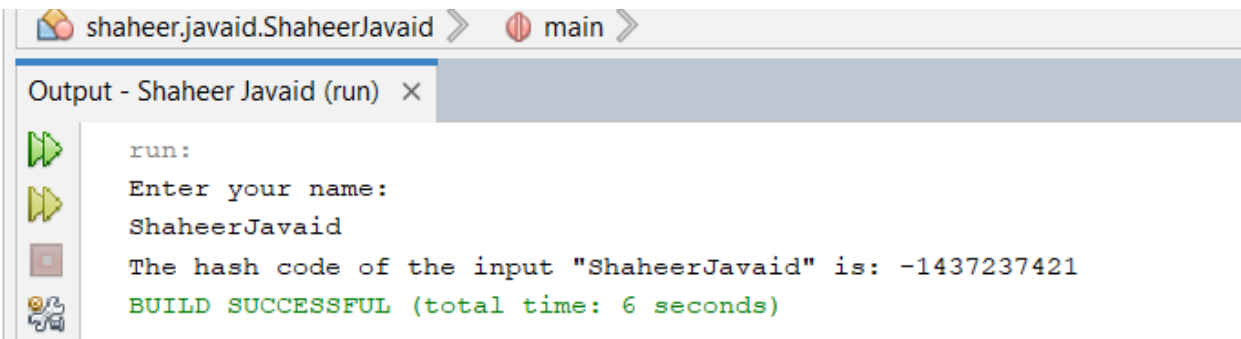
public class ShaheerJavaid {
    public static void main(String[] args) {
        Scanner scanner = new Scanner(System.in);

        System.out.println("Enter your name: ");
        String input = scanner.nextLine();

        int hashCode = input.hashCode();
        System.out.println("The hash code of the input \" + input + \" is: \" + hashCode);
    }
}
```

OUTPUT:

:



```
run:
Enter your name:
ShaheerJavaid
The hash code of the input "ShaheerJavaid" is: -1437237421
BUILD SUCCESSFUL (total time: 6 seconds)
```

TASK NO 3:

Create a java project, suppose you work for a company that needs to manage a list of employees. Each employee has a unique combination of a name and an ID. Your goal is to ensure that you can track employees effectively and avoid duplicate entries in your system. Requirements

a. Employee Class: You need to create an Employee class that includes:

- name: The employee's name (String).
- id: The employee's unique identifier (int)
- Override the hashCode() and equals() methods to ensure that two employees are considered equal if they have the same name and id.

b. Employee Management: You will use a HashSet to store employee records. This will help you avoid duplicate entries.

c. Operations: Implement operations to:

- Add new employees to the record.
- Check if an employee already exists in the records.
- Display all employees

.INPUT:

```
package shaheer.javaaid;
import java.util.HashSet;
import java.util.Scanner;

class Employee {
    private int id;
    private String name;

    public Employee(int id, String name) {
        this.id = id;
        this.name = name;
    }

    @Override
    public int hashCode() {
        return id + name.hashCode();
    }

    @Override
    public boolean equals(Object obj) {
        if (this == obj) return true;
        if (obj == null || getClass() != obj.getClass()) return false;
        Employee other = (Employee) obj;
        return id == other.id && name.equals(other.name);
    }

    @Override
    public String toString() {
        return "ID: " + id + ", Name: " + name;
    }
}
```

:

```
    }  
}  
  
public class ShaheerJavaid {  
    public static void main(String[] args) {  
        HashSet<Employee> employees = new HashSet<>();  
        Scanner scanner = new Scanner(System.in);  
        int choice;  
  
        do {  
            System.out.println("\n1. Add Employee");  
            System.out.println("2. Check Employee Existence");  
            System.out.println("3. Display All Employees");  
            System.out.println("4. Exit");  
            System.out.print("Enter your choice: ");  
            choice = scanner.nextInt();  
            scanner.nextLine();  
  
            switch (choice) {  
                case 1:  
                    System.out.print("Enter Employee ID: ");  
                    int id = scanner.nextInt();  
                    scanner.nextLine();  
                    System.out.print("Enter Employee Name: ");  
                    String name = scanner.nextLine();  
  
                    Employee newEmployee = new Employee(id, name);  
                    if (employees.add(newEmployee)) {  
                        System.out.println("Employee added successfully!");  
                    } else {  
                        System.out.println("Duplicate employee! Not added.");  
                    }  
                }  
            }  
        }  
    }  
}
```

```
        System.out.println("Duplicate employee! Not added.");
    }
    break;

    case 2:
        System.out.print("Enter Employee ID to check: ");
        int checkId = scanner.nextInt();
        scanner.nextLine(); // Consume newline
        System.out.print("Enter Employee Name to check: ");
        String checkName = scanner.nextLine();

        Employee checkEmployee = new Employee(checkId, checkName);
        if (employees.contains(checkEmployee)) {
            System.out.println("Employee exists in the records.");
        } else {
            System.out.println("Employee not found.");
        }
        break;

    case 3:
        if (employees.isEmpty()) {
            System.out.println("No employees in the records.");
        } else {
            System.out.println("Employee Records:");
            for (Employee emp : employees) {
                System.out.println(emp);
            }
        }
        break;

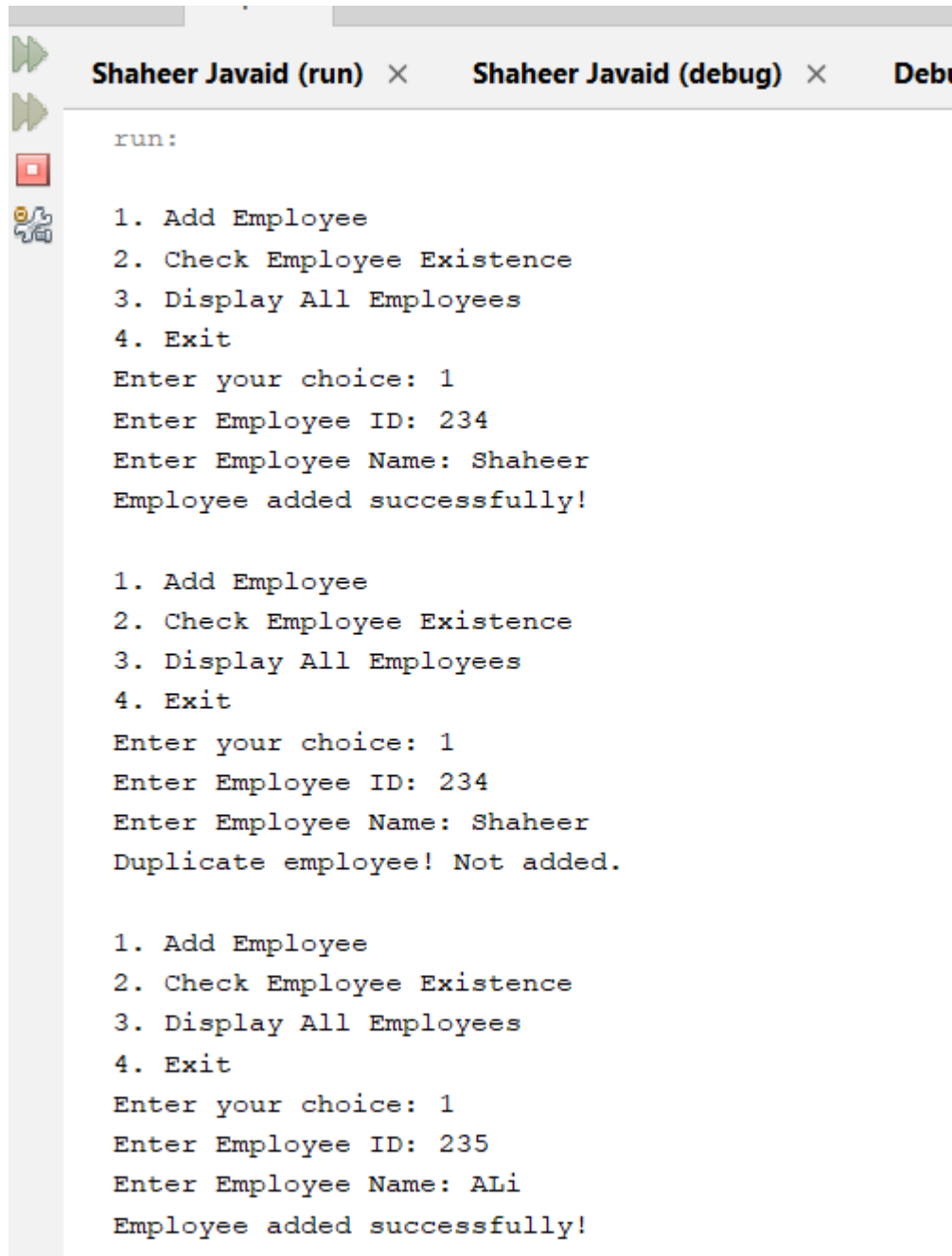
        break;

        case 4: // Exit
            System.out.println("Exiting...");
            break;

        default:
            System.out.println("Invalid choice! Try again.");
        }
    } while (choice != 4);

    scanner.close();
}
```

:

OUTPUT:

```
run:
1. Add Employee
2. Check Employee Existence
3. Display All Employees
4. Exit
Enter your choice: 1
Enter Employee ID: 234
Enter Employee Name: Shaheer
Employee added successfully!

1. Add Employee
2. Check Employee Existence
3. Display All Employees
4. Exit
Enter your choice: 1
Enter Employee ID: 234
Enter Employee Name: Shaheer
Duplicate employee! Not added.

1. Add Employee
2. Check Employee Existence
3. Display All Employees
4. Exit
Enter your choice: 1
Enter Employee ID: 235
Enter Employee Name: ALi
Employee added successfully!
```

TASK NO 4:

.Create a Color class that has red, green, and blue values. Two colors are considered equal if their RGB values are the same

INPUT:

```
-  */
package shaheer.javaaid;
class Color {
    private int red;
    private int green;
    private int blue;

    // Constructor to initialize the color with RGB values
] public Color(int red, int green, int blue) {
    this.red = red;
    this.green = green;
    this.blue = blue;
- }

    // Overriding equals() to compare colors by their RGB values
@Override
] public boolean equals(Object obj) {
    if (this == obj) return true; // If both are the same object
    if (obj == null || getClass() != obj.getClass()) return false; // If the
    Color other = (Color) obj;
    return red == other.red && green == other.green && blue == other.blue; /
- }

    // Overriding hashCode() to return a unique code for each color based on RGB
@Override
] public int hashCode() {
    return red * 31 + green * 17 + blue * 7; // Simple calculation to genera
- }
```


:

```
@Override
public String toString() {
    return "Color: RGB(" + red + ", " + green + ", " + blue + ")";
}

public class ShaheerJavaid {
    public static void main(String[] args) {
        Color color1 = new Color(255, 0, 0); // Red color
        Color color2 = new Color(255, 0, 0); // Red color

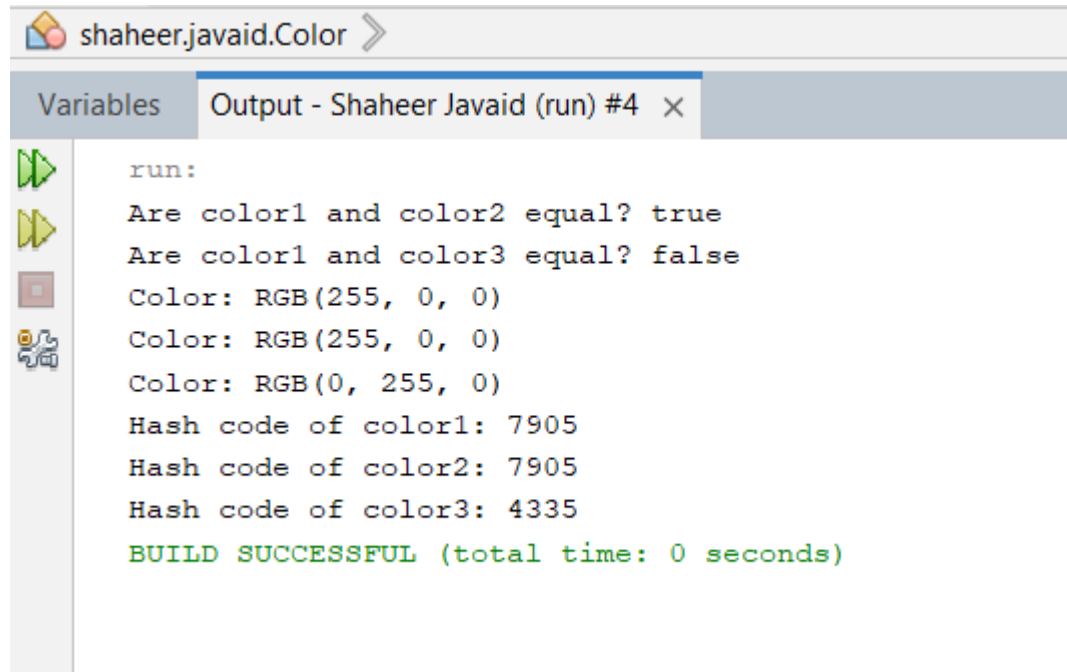
        // Create another color with different RGB values
        Color color3 = new Color(0, 255, 0); // Green color

        // Compare colors
        System.out.println("Are color1 and color2 equal? " + color1.equals(color2)); /
        System.out.println("Are color1 and color3 equal? " + color1.equals(color3)); /

        // Display the colors
        System.out.println(color1); // Color: RGB(255, 0, 0)
        System.out.println(color2); // Color: RGB(255, 0, 0)
        System.out.println(color3); // Color: RGB(0, 255, 0)

        // Display hash codes
        System.out.println("Hash code of color1: " + color1.hashCode());
        System.out.println("Hash code of color2: " + color2.hashCode());
        System.out.println("Hash code of color3: " + color3.hashCode());
    }
}
```

OUTPUT:



The image shows a screenshot of an IDE's output window. At the top, the file path 'shaheer.java:Color' is displayed with a right-pointing arrow. Below this, there are two tabs: 'Variables' and 'Output - Shaheer Javaid (run) #4'. The 'Output' tab is active. On the left side of the output area, there is a vertical toolbar with icons for running (green play button), stepping through (yellow play button), stopping (red square), and debugging (bug icon). The output text is as follows:

```
run:
Are color1 and color2 equal? true
Are color1 and color3 equal? false
Color: RGB(255, 0, 0)
Color: RGB(255, 0, 0)
Color: RGB(0, 255, 0)
Hash code of color1: 7905
Hash code of color2: 7905
Hash code of color3: 4335
BUILD SUCCESSFUL (total time: 0 seconds)
```