LAB NO 2
ArrayList and Vector in JAVA

OBJECTIVE: To implement ArrayList and Vector

Question 1:

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

INPUT:

```
import java.util.Vector;
public class JavaApp {

    public static void main(String[] args) {

        Vector<Integer> numbers = new Vector<>();
        numbers.add(5);
        numbers.add(19);
        numbers.add(5);
        numbers.add(8);
        numbers.add(89);
        numbers.add(69);
        numbers.add(16);
        numbers.add(4);
        numbers.add(10);
        numbers.add(6);
        System.out.println("The integers  are:");
        for (int num : numbers) {
            System.out.println(num);
        }

        int sum = 0;
        for (int num : numbers) {
            sum += num;
        }
        System.out.println("The sum of the nymber is: " + sum);
    }
}
```

OUTPUT:

```
The integers  are:
5
19
5
8
89
69
16
4
10
6
The sum of the nymber is: 231
```

Question 2:

Create a ArrayList of string. Write a menu driven program which:

a. Displays all the elements

b. Displays the largest String

INPUT:

```
import java.util.Scanner;

public class JavaApp {
    public static void main(String[] args) {
        ArrayList<String> stringList = new ArrayList<>();
        stringList.add("Biryani");
        stringList.add("nehari");
        stringList.add("pasta");
        stringList.add("double chesse burger");
        stringList.add("loki ki bhojiya");
        Scanner scanner = new Scanner(System.in);
        while (true) {
            System.out.println("\nMenu:");
            System.out.println("1. Display all elements");
            System.out.println("2. Display the largest string");
            System.out.println("3. Exit");
            System.out.print("Choose an option: ");
            int choice = scanner.nextInt();
            scanner.nextLine();
            if (choice == 1) {
                System.out.println("\nAll elements in the ArrayList:");
                for (String str : stringList) {
                    System.out.println(str);
                }
            }
            else if (choice == 2) {
               String largest = stringList.get(4);
                for (String str : stringList) {
                    if (str.compareTo(largest) > 0) {
                        largest = str;
                    }
                }
                System.out.println("\nThe largest string is: " + largest);
            }

        }

    }
```

OUTPUT:

```
Menu:
1. Display all elements
2. Display the largest string
3. Exit
Choose an option:
1

All elements in the ArrayList:
Biryani
nehari
pasta
double chesse burger
loki ki bhojiya
```

Question 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:

```
import java.util.ArrayList;
import java.util.*;


class Employee implements Comparable<Employee> {
    private final int empId;
    private final String empName;
    private final String empGender;
    private final int yearOfJoining;

    public Employee(int empId, String empName, String empGender, int yearOfJoining) {
        this.empId = empId;
        this.empName = empName;
        this.empGender = empGender;
        this.yearOfJoining = yearOfJoining;
    }
    public int getEmpId() {
        return empId;
    }
    public String getEmpName() {
        return empName;
    }
    public String getEmpGender() {
        return empGender;
    }
    public int getYearOfJoining() {
        return yearOfJoining;
    }
    @Override
    public int compareTo(Employee other) {
        return Integer.compare(this.yearOfJoining, other.yearOfJoining);
    }
    @Override
    public String toString() {
        return "ID: " + empId + ", Name: " + empName + ", Gender: " + empGender + ", Year of Joining: " + yearOfJoining;
    }
}
```

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

        ArrayList<Employee> employeeList = new ArrayList<>();


        employeeList.add(new Employee(101, "Ali", "Male", 2018));
        employeeList.add(new Employee(102, "ayesha", "Female", 2023));
        employeeList.add(new Employee(103, "Ahmed", "Male", 2022));
        employeeList.add(new Employee(104, "saima", "Female", 2010));
        employeeList.add(new Employee(105, "anas", "Male", 2014));

        System.out.println("Employees before sorting:");
        for (Employee emp : employeeList) {
            System.out.println(emp);
        }
        Collections.sort(employeeList);
        System.out.println("\nEmployees sorted by Year of Joining (Comparable):");
        for (Employee emp : employeeList) {
            System.out.println(emp);
        }
        Collections.sort(employeeList, new Comparator<Employee>() {
            @Override
            public int compare(Employee e1, Employee e2) {
                return e1.getEmpName().compareTo(e2.getEmpName()); // Sort by Name
            }
        });
        System.out.println("\nEmployees sorted by Name (Comparator):");
        for (Employee emp : employeeList) {
            System.out.println(emp);
        }
    }
}
```

OUTPUT:

```
Employees before sorting:
ID: 101, Name: Ali, Gender: Male, Year of Joining: 2018
ID: 102, Name: ayesha, Gender: Female, Year of Joining: 2023
ID: 103, Name: Ahmed, Gender: Male, Year of Joining: 2022
ID: 104, Name: saima, Gender: Female, Year of Joining: 2010
ID: 105, Name: anas, Gender: Male, Year of Joining: 2014

Employees sorted by Year of Joining (Comparable):
ID: 104, Name: saima, Gender: Female, Year of Joining: 2010
ID: 105, Name: anas, Gender: Male, Year of Joining: 2014
ID: 101, Name: Ali, Gender: Male, Year of Joining: 2018
ID: 103, Name: Ahmed, Gender: Male, Year of Joining: 2022
ID: 102, Name: ayesha, Gender: Female, Year of Joining: 2023

Employees sorted by Name (Comparator):
ID: 103, Name: Ahmed, Gender: Male, Year of Joining: 2022
ID: 101, Name: Ali, Gender: Male, Year of Joining: 2018
ID: 105, Name: anas, Gender: Male, Year of Joining: 2014
ID: 102, Name: ayesha, Gender: Female, Year of Joining: 2023
ID: 104, Name: saima, Gender: Female, Year of Joining: 2010
--------------------------------------------------------------------------------
```

Question 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:

```
import java.util.Collections;
import java.util.Vector;

public class JavaApp {

    public static void main(String[] args) {

        Vector<Integer> numbers = new Vector<>();
        numbers.add(5);
        numbers.add(19);
        numbers.add(5);
        numbers.add(8);
        numbers.add(89);
        numbers.add(69);
        numbers.add(16);
        numbers.add(4);
        numbers.add(10);
        numbers.add(6);
        System.out.println("The integers  are:");
        for (int num : numbers) {
            System.out.println(num);
        }

        int sum = 0;
        for (int num : numbers) {
            sum += num;
        }
        System.out.println("The sum of the nymber is: " + sum);
         int maxElement = Collections.max(numbers);
        System.out.println("\nMaximum element in the Vector: " + maxElement);
    }
}
```

OUTPUT:

```
The integers  are:
5
19
5
8
89
69
16
4
10
6
The sum of the nymber is: 231

Maximum element in the Vector: 89
```

Question 5:

Find the k-th smallest element in a sorted ArrayList

INPUT:

```java
public class JavaApp {

    public static void main(String[] args) {
       ArrayList<Integer> sortedList = new ArrayList<>(Arrays.asList(1, 3, 5, 7, 9, 11, 13, 15, 17, 19));

        System.out.println("Sorted ArrayList: " + sortedList);

        int k = 4;
        if (k > 0 && k <= sortedList.size()) {
            int kthSmallest = sortedList.get(k - 1);
            System.out.println("The " + k + "-th smallest element is: " + kthSmallest);
        } else {
            System.out.println("Invalid value of k. Please ensure that k is between 1 and " + sortedList.size());
        }
    }
}
```

OUTPUT:

```
--- exec:3.1.0:exec (default-cli) @ javaApp ---
Sorted ArrayList: [1, 3, 5, 7, 9, 11, 13, 15, 17, 19]
The 4-th smallest element is: 7
```

Question 6:

Write a program to merge two ArrayLists into one

INPUT:

```
import java.util.ArrayList;
import java.util.Arrays;


public class JavaApp {

    public static void main(String[] args) {

        ArrayList<Integer> list1 = new ArrayList<>(Arrays.asList(1, 2, 3, 4, 5));
        ArrayList<Integer> list2 = new ArrayList<>(Arrays.asList(6, 7, 8, 9, 10));
        System.out.println("List 1: " + list1);
        System.out.println("List 2: " + list2);
        list1.addAll(list2);
        System.out.println("\nMerged List: " + list1);
    }
}
```

OUTPUT:

```
List 1: [1, 2, 3, 4, 5]
List 2: [6, 7, 8, 9, 10]

Merged List: [1, 2, 3, 4, 5, 6, 7, 8, 9, 10]
```

HOME TASKS

Question 1:

1. Create a Vector storing integer objects as an input. a. Sort the vector b. Display largest number c. Display smallest number

INPUT:

```
public class JavaApp {

    public static void main(String[] args) {


        Vector<Integer> vector = new Vector<>();
        Scanner scanner = new Scanner(System.in);
        System.out.println("Enter 5 integers: ");
        for (int i = 0; i < 5; i++) {
            vector.add(scanner.nextInt());
        }
        System.out.println("Original Vector: " +vector);
        Collections.sort(vector);
        System.out.println("\nSorted Vector: " + vector);

        int largest = Collections.max(vector);
        System.out.println("\nLargest number: " + largest);
        int smallest = Collections.min(vector);
        System.out.println("Smallest number: " + smallest);

        scanner.close();
    }
}
```

OUTPUT:

```
Enter 5 integers:
45
67
89
56
3
Original Vector: [45, 67, 89, 56, 3]

Sorted Vector: [3, 45, 56, 67, 89]

Largest number: 89
Smallest number: 3
```

Question 2:

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

INPUT:

```
public class JavaApp {

    public static void main(String[] args) {

        Scanner scanner = new Scanner(System.in);

        System.out.println("Enter a string:");
        String userInput = scanner.nextLine();

        int hashCode = userInput.hashCode();

        System.out.println("Hash code of the entered string: " + hashCode);
        scanner.close();
    }
}
```

OUTPUT:

```
Enter a string:
ANAS
Hash code of the entered string: 2013471
------------------------------------------
```

Question 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:

```
import java.util.HashSet;
import java.util.Scanner;
import java.util.Set;

class Employee {
    String name;
    int id;

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

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

    @Override
    public int hashCode() {
        return 31 * name.hashCode() + id;
    }

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

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

        Set<Employee> employees = new HashSet<>();
        Scanner scanner = new Scanner(System.in);

        while (true) {
            System.out.println("\n1. Add Employee");
            System.out.println("2. Check if Employee Exists");
            System.out.println("3. Display All Employees");
            System.out.println("4. Exit");
            System.out.print("Choose an option: ");
```

```
int choice = scanner.nextInt();
scanner.nextLine();

if (choice == 1) {

    System.out.print("Enter employee name: ");
    String name = scanner.nextLine();
    System.out.print("Enter employee ID: ");
    int id = scanner.nextInt();
    scanner.nextLine();

    Employee employee = new Employee(name, id);
    if (employees.add(employee)) {
        System.out.println("Employee added successfully!");
    } else {
        System.out.println("Employee already exists!");
    }

} else if (choice == 2) {

    System.out.print("Enter employee name to check: ");
    String name = scanner.nextLine();
    System.out.print("Enter employee ID to check: ");
    int id = scanner.nextInt();
    scanner.nextLine();

    Employee employee = new Employee(name, id);
    if (employees.contains(employee)) {
        System.out.println("Employee exists!");
    } else {
        System.out.println("Employee not found!");
    }

} else if (choice == 3) {

    if (employees.isEmpty()) {
        System.out.println("No employees to display.");
    } else {
        System.out.println("Employee List:");
        for (Employee emp : employees) {
            System.out.println(emp);
        }
    }

} else if (choice == 4) {

    System.out.println("Exiting...");
    break;
} else {
    System.out.println("Invalid choice. Please try again.");
}

scanner.close();
```

OUTPUT:

```
1. Add Employee
2. Check if Employee Exists
3. Display All Employees
4. Exit
Choose an option: 1
Enter employee name: anas
Enter employee ID: 101
Employee added successfully!

1. Add Employee
2. Check if Employee Exists
3. Display All Employees
4. Exit
Choose an option: 1
Enter employee name: ermin
Enter employee ID: 102
Employee added successfully!

1. Add Employee
2. Check if Employee Exists
3. Display All Employees
4. Exit
Choose an option: 3
Employee List:
Name: ermin, ID: 102
Name: anas, ID: 101
```