LAB 02

To implement Array List and Vector.

OBJECTIVE: To implement Array List and Vector.

LAB TASKS

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

INPUT:

```
package lab.pkg02;
import java.util.Vector;
public class Lab02 {
    public static void main(String[] args) {
        Vector<Integer> numbers = new Vector<>();
        for (int i = 1; i <= 10; i++) {
            numbers.add(e: i);
        }
        System.out.println("Integers in the Vector: " + numbers);
        int sum = 0;
        for (int num : numbers) {
            sum += num;
        }
        System.out.println("Sum of the integers: " + sum);
    }
}
```

OUTPUT:

```
Integers in the Vector: [1, 2, 3, 4, 5, 6, 7, 8, 9, 10]
Sum of the integers: 55
```

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.*;
public class Lab02 {
    public static void main (String[] args) {
        Scanner Sc = new Scanner(source: System.in);
        ArrayList<String> menu = new ArrayList<String>();

        System.out.println(x: "Enter the length of your ArrayList:");
        int x = Sc.nextInt();

        for (int i = 0; i < x; i++) {
            System.out.print("Enter item " + (i + 1) + ": ");
            String a = Sc.next();
            menu.add(e: a);
        }
        System.out.print("MENU: " + menu);
        System.out.println();

        String longest_string = "";
        for (String b : menu) {
            if (b.length() > longest_string.length()) {
                longest_string = b;
            }
        }
        System.out.println("Longest item in the list is: " + longest_string);
        Sc.close();
    }
}
```

OUTPUT:

```
Enter the length of your ArrayList:
3
Enter item 1: biryani
Enter item 2: apple
Enter item 3: mango
MENU: [biryani, apple, mango]
Longest item in the list is: biryani
```

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.Collections;
import java.util.Comparator;

class Employee implements Comparable<Employee> {
    int empId;
    String empName;
    String empGender;
    int yearOfJoining;
    double empSalary;

    public Employee(int empId, String empName, String empGender, int yearOfJoining, double empSalary) {
        this.empId = empId;
        this.empName = empName;
        this.empGender = empGender;
        this.yearOfJoining = yearOfJoining;
        this.empSalary = empSalary;
    }
    public int getYearOfJoining() {
        return yearOfJoining;
    }
    @Override
    public String toString() {
        return "Employee [ID=" + empId + ", Name=" + empName + ", Gender=" + empGender
            + ", Year of Joining=" + yearOfJoining + ", Salary=" + empSalary + "]";
    }
    @Override
    public int compareTo(Employee other) {
        return Integer.compare(x: this.yearOfJoining, y: other.yearOfJoining);
    }
}
```

```
class YearOfJoiningComparator implements Comparator<Employee> {
    @Override
    public int compare(Employee e1, Employee e2) {
        return Integer.compare(x: e1.getYearOfJoining(), y: e2.getYearOfJoining());
    }
}
public class Lab02 {
    public static void main(String[] args) {
        ArrayList<Employee> employees = new ArrayList<>();

        employees.add(new Employee(empId: 1, empName: "Zayyan", empGender:"Male", yearOfJoining: 2020, empSalary:65000));
        employees.add(new Employee(empId: 2, empName: "Ahmed", empGender:"Male", yearOfJoining: 2018, empSalary:23000));
        employees.add(new Employee(empId: 3, empName: "Fahad", empGender:"Male", yearOfJoining: 2012, empSalary:84000));
        employees.add(new Employee(empId: 4, empName: "Zara", empGender:"Female", yearOfJoining: 2020, empSalary:77000));
        employees.add(new Employee(empId: 5, empName: "Zunaira", empGender:"Female", yearOfJoining: 2014, empSalary:54000));

        Collections.sort(list:employees);
        System.out.println(x: "Sorted by Year of Joining (using Comparable):");
        for (Employee e : employees) {
            System.out.println(x: e);
        }
        Collections.sort(list:employees, new YearOfJoiningComparator());
        System.out.println(x: "\nSorted by Year of Joining (using Comparator):");
        for (Employee e : employees) {
            System.out.println(x: e);
        }
    }
}
```

OUTPUT:

```
Sorted by Year of Joining (using Comparable):
Employee [ID=3, Name=Fahad, Gender=Male, Year of Joining=2012, Salary=84000.0]
Employee [ID=5, Name=Zunaira, Gender=Female, Year of Joining=2014, Salary=54000.0]
Employee [ID=2, Name=Ahmed, Gender=Male, Year of Joining=2018, Salary=23000.0]
Employee [ID=1, Name=Zayyan, Gender=Male, Year of Joining=2020, Salary=65000.0]
Employee [ID=4, Name=Zara, Gender=Female, Year of Joining=2020, Salary=77000.0]

Sorted by Year of Joining (using Comparator):
Employee [ID=3, Name=Fahad, Gender=Male, Year of Joining=2012, Salary=84000.0]
Employee [ID=5, Name=Zunaira, Gender=Female, Year of Joining=2014, Salary=54000.0]
Employee [ID=2, Name=Ahmed, Gender=Male, Year of Joining=2018, Salary=23000.0]
Employee [ID=1, Name=Zayyan, Gender=Male, Year of Joining=2020, Salary=65000.0]
Employee [ID=4, Name=Zara, Gender=Female, Year of Joining=2020, Salary=77000.0]
```

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.Vector;
public class Lab02 {
    public static void main(String[] args) {
        Vector<Integer> numbers = new Vector<>();
        numbers.add(e: 10);
        numbers.add(e: 20);
        numbers.add(e: 30);
        numbers.add(e: 40);
        numbers.add(e: 50);
        numbers.add(e: 60);
        numbers.add(e: 70);
        numbers.add(e: 80);
        numbers.add(e: 90);
        numbers.add(e: 100);

        System.out.println(x: "Integers in the Vector:");
        for (int num : numbers) {
            System.out.print(num + " ");
        }
        System.out.println();
        int sum = 0;
        for (int num : numbers) {
            sum += num;
        }
        System.out.println("Sum of the integers: " + sum);
        int max = numbers.get(index: 0);
        for (int num : numbers) {
            if (num > max) {
                max = num;
            }
        }
        System.out.println("Maximum element in the Vector: " + max);
    }
}
```

OUTPUT:

```
Integers in the Vector:
10 20 30 40 50 60 70 80 90 100
Sum of the integers: 550
Maximum element in the Vector: 100
```

5. Find the k-th smallest element in a sorted ArrayList

INPUT:

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

public class Lab02 {
    public static void main(String[] args) {
        ArrayList<Integer> sortedList = new ArrayList<>(c: Arrays.asList(a: 1, a: 3, a: 5, a: 7, a: 9, a: 11, a: 13, a: 15, a: 17, a: 19));

        Scanner scanner = new Scanner(source: System.in);
        System.out.print("Enter the value of k (1 to " + sortedList.size() + "): ");
        int k = scanner.nextInt();
        if (k < 1 || k > sortedList.size()) {
            System.out.println("Invalid value for k. Please enter a number between 1 and " + sortedList.size() + ".");
        } else {
            int kthSmallest = sortedList.get(k - 1);
            System.out.println("The " + k + "-th smallest element is: " + kthSmallest);
        }
        scanner.close();
    }
}
```

OUTPUT:

```
Enter the value of k (1 to 10): 8
The 8-th smallest element is: 15
```

6. Write a program to merge two ArrayLists into one

INPUT:

```
import java.util.ArrayList;
import java.util.Arrays;
public class Lab02 {
    public static void main(String[] args) {
        ArrayList<String> list1 = new ArrayList<>(c: Arrays.asList(a: "Apple", a: "Banana", a: "Cherry"));
        ArrayList<String> list2 = new ArrayList<>(c: Arrays.asList(a: "Date", a: "Fig", a: "Grape"));

        System.out.println("List 1: " + list1);
        System.out.println("List 2: " + list2);

        ArrayList<String> mergedList = new ArrayList<>(c: list1);
        mergedList.addAll(c: list2);
        System.out.println("Merged List: " + mergedList);
    }
}
```

OUTPUT:

```
List 1: [Apple, Banana, Cherry]
List 2: [Date, Fig, Grape]
Merged List: [Apple, Banana, Cherry, Date, Fig, Grape]
```

HOME TASK

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

INPUT:

```
import java.util.Scanner;
import java.util.Vector;
import java.util.Collections;
public class Lab02{
    public static void main(String[] args) {
        Scanner scanner = new Scanner(source: System.in);
        Vector<Integer> numbers = new Vector<>();

        System.out.println(x: "Enter integers to store in the Vector (type 'done' to finish):");

        while (scanner.hasNext()) {
            if (scanner.hasNextInt()) {
                numbers.add(e: scanner.nextInt());
            } else if (scanner.next().equalsIgnoreCase(anotherString: "done")) {
                break;
            } else {
                System.out.println(x: "Invalid input. Please enter integers or type 'done' to finish.");
            }
        }
        Collections.sort(list:numbers);
        System.out.println("Sorted Vector: " + numbers);

        if (numbers.isEmpty()) {
            System.out.println(x: "The Vector is empty. No largest or smallest number to display.");
        } else {
            int largest = numbers.lastElement();
            int smallest = numbers.firstElement();

            System.out.println("Largest number: " + largest);
            System.out.println("Smallest number: " + smallest);
        }
        scanner.close();
    }
}
```

OUTPUT:

```
Enter integers to store in the Vector (type 'done' to finish):
8
2
5
8
4
done
Sorted Vector: [2, 4, 5, 8, 8]
Largest number: 8
Smallest number: 2
```

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

INPUT:

```
import java.util.Scanner;
public class Lab02 {
    public static void main(String[] args) {
        Scanner scanner = new Scanner(source: System.in);
        System.out.println(x: "Enter a string to get its hash code (type 'exit' to finish):");
        while (true) {
            String input = scanner.nextLine();
            if (input.equalsIgnoreCase(anotherString: "exit")) {
                break;
            }
            int hashCode = input.hashCode();
            System.out.println("Hash code for \"" + input + "\": " + hashCode);
        }
        scanner.close();
        System.out.println(x: "Program terminated.");
    }
}
```

OUTPUT:

```
Enter a string to get its hash code (type 'exit' to finish):
superb
Hash code for "superb": -891125689
```

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.ArrayList;
import java.util.List;
import java.util.Scanner;

public class Lab02{
    public static class Employee {
        private String id;
        private String name;

        public Employee(String id, String name) {
            this.id = id;
            this.name = name;
        }
        public String getId() {
            return id;
        }
        public String getName() {
            return name;
        }
        @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.equals(anObject: employee.id) && name.equals(anObject: employee.name);
        }
        @Override
        public int hashCode() {
            return 31 * id.hashCode() + name.hashCode();
        }
```

```
    @Override
    public String toString() {
        return "Employee{" +
                "id='" + id + '\'' +
                ", name='" + name + '\'' +
                '}';
    }
}
public static class EmployeeManager {
    private List<Employee> employees;

    public EmployeeManager() {
        this.employees = new ArrayList<>();
    }
    public boolean addEmployee(Employee employee) {
        if (employees.contains(o: employee)) {
            System.out.println("Employee with ID " + employee.getId() + " already exists.");
            return false;
        }
        employees.add(e: employee);
        System.out.println("Employee added: " + employee);
        return true;
    }
    public void displayEmployees() {
        if (employees.isEmpty()) {
            System.out.println(x: "No employees to display.");
            return;
        }
        System.out.println(x: "Employee List:");
        for (Employee employee : employees) {
            System.out.println(x: employee);
        }
    }
}
```

```
    public static void main(String[] args) {
        Scanner scanner = new Scanner(source: System.in);
        EmployeeManager employeeManager = new EmployeeManager();

        while (true) {
            System.out.println(x: "\nEmployee Management System");
            System.out.println(x: "1. Add Employee");
            System.out.println(x: "2. Display Employees");
            System.out.println(x: "3. Exit");
            System.out.print(s: "Choose an option: ");
            int choice = scanner.nextInt();
            scanner.nextLine(); // Consume newline

            switch (choice) {
                case 1:
                    System.out.print(s: "Enter Employee ID: ");
                    String id = scanner.nextLine();
                    System.out.print(s: "Enter Employee Name: ");
                    String name = scanner.nextLine();
                    Employee employee = new Employee(id, name);
                    employeeManager.addEmployee(employee);
                    break;
                case 2:
                    employeeManager.displayEmployees();
                    break;
                case 3:
                    System.out.println(x: "Exiting the program.");
                    scanner.close();
                    return;
                default:
                    System.out.println(x: "Invalid choice. Please try again.");
            }
        }
    }
}
```

OUTPUT:

```
Employee Management System
1. Add Employee
2. Display Employees
3. Exit
Choose an option: 1
Enter Employee ID: 123
Enter Employee Name: Laiba
Employee added: Employee{id='123', name='Laiba'}

Employee Management System
1. Add Employee
2. Display Employees
3. Exit
Choose an option: 2
Employee List:
Employee{id='123', name='Laiba'}

Employee Management System
1. Add Employee
2. Display Employees
3. Exit
```

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:

```
public class Lab02 {
    public static void main(String[] args) {
        Color color1 = new Color(red: 255, green: 0, blue:0); // Red
        Color color2 = new Color(red: 0, green: 255, blue:0); // Green
        Color color3 = new Color(red: 0, green: 0, blue:255); // Blue
        Color color4 = new Color(red: 255, green: 0, blue:0); // Another Red

        System.out.println("Color 1: " + color1);
        System.out.println("Color 2: " + color2);
        System.out.println("Color 3: " + color3);
        System.out.println("Color 4: " + color4);

        System.out.println("Color 1 equals Color 2: " + color1.equals(obj: color2));
        System.out.println("Color 1 equals Color 4: " + color1.equals(obj: color4));
    }
    static class Color {
        public int red;
        public int green;
        public int blue;

        public Color(int red, int green, int blue) {
            this.red = red;
            this.green = green;
            this.blue = blue;
        }
        public int getRed() {
            return red;
        }
        public int getGreen() {
            return green;
        }
        public int getBlue() {
            return blue;
        }
```

OUTPUT:

```
Color 1: Color{red=255, green=0, blue=0}
Color 2: Color{red=0, green=255, blue=0}
Color 3: Color{red=0, green=0, blue=255}
Color 4: Color{red=255, green=0, blue=0}
Color 1 equals Color 2: false
Color 1 equals Color 4: true
```