**SVKM’s NMIMS**

**Mukesh Patel School of Technology Management & Engineering**

**Computer Engineering Department**

Program: BTech Integrated, Semester IV

**Course: Java Programming**

**Experiment No.09**

**Part B**

|  |  |
| --- | --- |
| **Roll no.: A073** | **Name: Aryan Srivastava** |
| **Class: MCA Sem1** | **Batch: B3** |
| **Date of Experiment:** | **Date of Submission:** |
| **Grade:** |  |

1. **Program scenario and Program code:**

**Task1**  
import java.util.ArrayList;

import java.util.Scanner;

public class StudentMarks {

    public static void main(String[] args) {

        ArrayList<Integer> marks = new ArrayList<>();

        Scanner scanner = new Scanner(System.in);

        int choice;

        do {

            System.out.println("1. Add marks");

            System.out.println("2. Display total number of students");

            System.out.println("3. Display marks of a particular student");

            System.out.println("4. Display marks of all students");

            System.out.println("5. Delete marks of a particular student");

            System.out.println("6. Exit");

            System.out.print("Enter your choice: ");

            choice = scanner.nextInt();

            switch (choice) {

                case 1:

                    System.out.print("Enter marks of student: ");

                    marks.add(scanner.nextInt());

                    break;

                case 2:

                    System.out.println("Total number of students: " + marks.size());

                    break;

                case 3:

                    System.out.print("Enter student index");

                    int index = scanner.nextInt();

                    if (index >= 0 && index < marks.size()) {

                        System.out.println("Marks of student " + index + ": " + marks.get(index));

                    } else {

                        System.out.println("Invalid index!");

                    }

                    break;

                case 4:

                    System.out.println("Marks of all students: " + marks);

                    break;

                case 5:

                    System.out.print("Enter student index");

                    int deleteIndex = scanner.nextInt();

                    if (deleteIndex >= 0 && deleteIndex < marks.size()) {

                        marks.remove(deleteIndex);

                        System.out.println("Marks deleted for student " + deleteIndex);

                    } else {

                        System.out.println("Invalid index!");

                    }

                    break;

                case 6:

                    System.out.println("Exiting");

                    break;

                default:

                    System.out.println("Invalid choice");

            }

        } while (choice != 6);

        scanner.close();

    }

}

**Task2**

import java.util.ArrayList;

import java.util.Collections;

import java.util.Comparator;

import java.util.Scanner;

class Student {

    int rollNo;

    String name;

    String address;

    public Student(int rollNo, String name, String address) {

        this.rollNo = rollNo;

        this.name = name;

        this.address = address;

    }

    public String getDetails() {

        return "roll No: " + rollNo + ", mame: " + name + ", address: " + address;

    }

}

class NameComparator implements Comparator<Student> {

    public int compare(Student s1, Student s2) {

        return s1.name.compareTo(s2.name);

    }

}

public class StudentDetails {

    public static void main(String[] args) {

        ArrayList<Student> students = new ArrayList<>();

        Scanner scanner = new Scanner(System.in);

        char choice;

        do {

            System.out.print("Enter roll no: ");

            int rollNo = scanner.nextInt();

            scanner.nextLine();

            System.out.print("Enter name: ");

            String name = scanner.nextLine();

            System.out.print("Enter address: ");

            String address = scanner.nextLine();

            students.add(new Student(rollNo, name, address));

            System.out.print("Do you want to add another student? (y/n): ");

            choice = scanner.next().charAt(0);

        } while (choice == 'y' || choice == 'Y');

        Collections.sort(students, new NameComparator());

        System.out.println("Students sorted by name:");

        for (int i = 0; i < students.size(); i++) {

            System.out.println(students.get(i).getDetails());

        }

        scanner.close();

    }

}

**Task3**

import java.util.LinkedList;

import java.util.Scanner;

public class ItemList {

    public static void main(String[] args) {

        LinkedList<String> items = new LinkedList<>();

        Scanner scanner = new Scanner(System.in);

        int choice;

        do {

            System.out.println("1. Insert item");

            System.out.println("2. Remove item");

            System.out.println("3. Display items in ascending order");

            System.out.println("4. Display items in descending order");

            System.out.println("5. Display a particular item");

            System.out.println("6. Exit");

            System.out.print("Enter your choice: ");

            choice = scanner.nextInt();

            scanner.nextLine();

            switch (choice) {

                case 1:

                    System.out.print("Enter item name: ");

                    String itemName = scanner.nextLine();

                    System.out.print("Insert at (1: beginning, 2: end, 3: position): ");

                    int position = scanner.nextInt();

                switch (position) {

                    case 1:

                        items.addFirst(itemName);

                        break;

                    case 2:

                        items.addLast(itemName);

                        break;

                    case 3:

                        System.out.print("Enter position: ");

                        int pos = scanner.nextInt();

                        items.add(pos, itemName);

                        break;

                    default:

                        System.out.println("Invalid option!");

                        break;

                }

                    break;

                case 2:

                    System.out.print("Remove from (1: beginning, 2: end, 3: position): ");

                    int removePosition = scanner.nextInt();

                switch (removePosition) {

                    case 1:

                        items.removeFirst();

                        break;

                    case 2:

                        items.removeLast();

                        break;

                    case 3:

                        System.out.print("Enter position: ");

                        int pos = scanner.nextInt();

                        items.remove(pos);

                        break;

                    default:

                        System.out.println("Invalid option!");

                        break;

                }

                    break;

                case 3:

                    System.out.println("Items in ascending order: " + items);

                    break;

                case 4:

                    System.out.println("Items in descending order: ");

                    for (int i = items.size() - 1; i >= 0; i--) {

                        System.out.println(items.get(i));

                    }

                    break;

                case 5:

                    System.out.print("Enter index of item to display: ");

                    int index = scanner.nextInt();

                    if (index >= 0 && index < items.size()) {

                        System.out.println("Item at index " + index + ": " + items.get(index));

                    } else {

                        System.out.println("Invalid index!");

                    }

                    break;

                case 6:

                    System.out.println("Exit");

                    break;

                default:

                    System.out.println("Invalid choice! Please try again.");

            }

        } while (choice != 6);

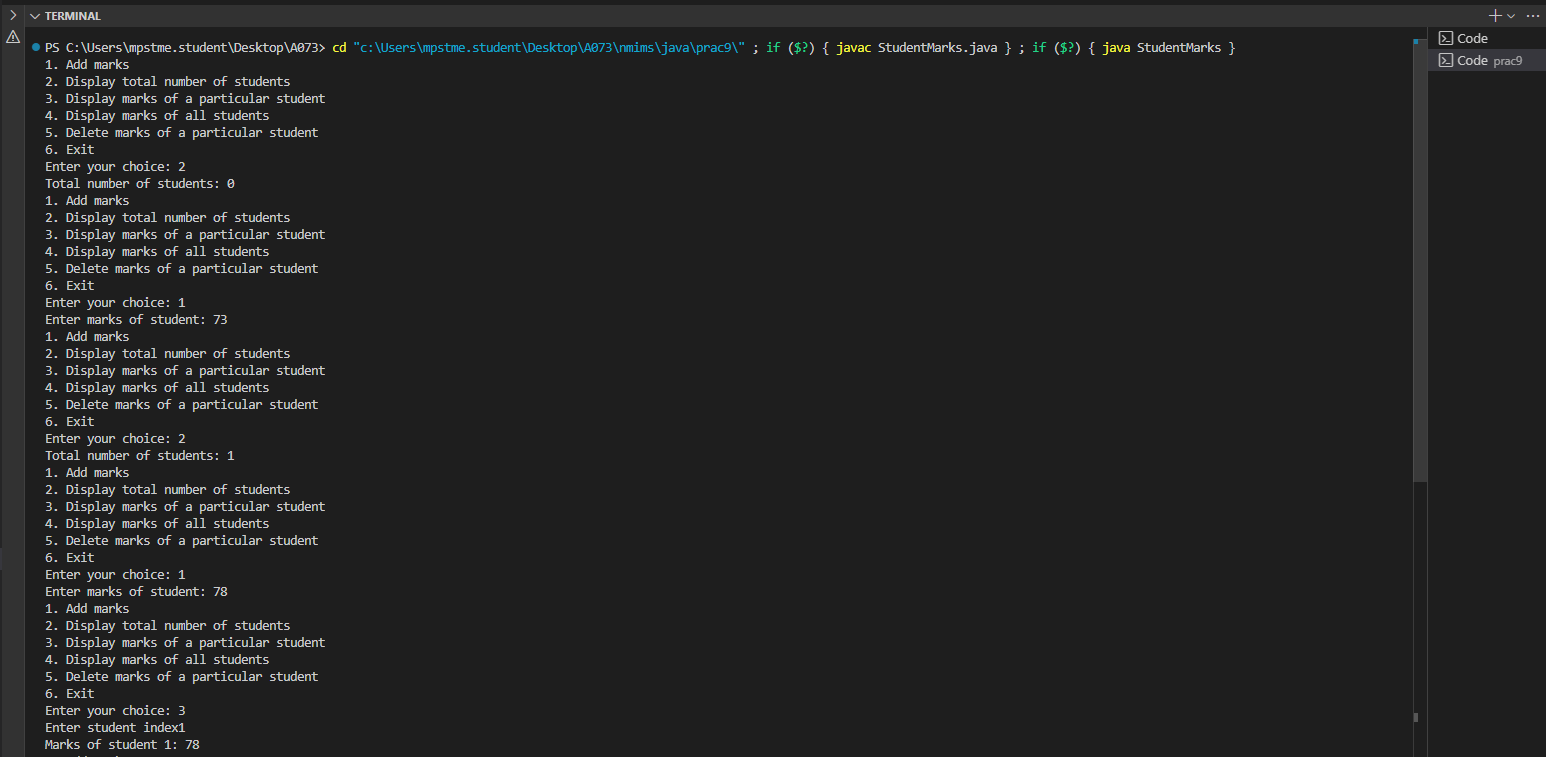
        scanner.close();

    }

}

1. **Output**

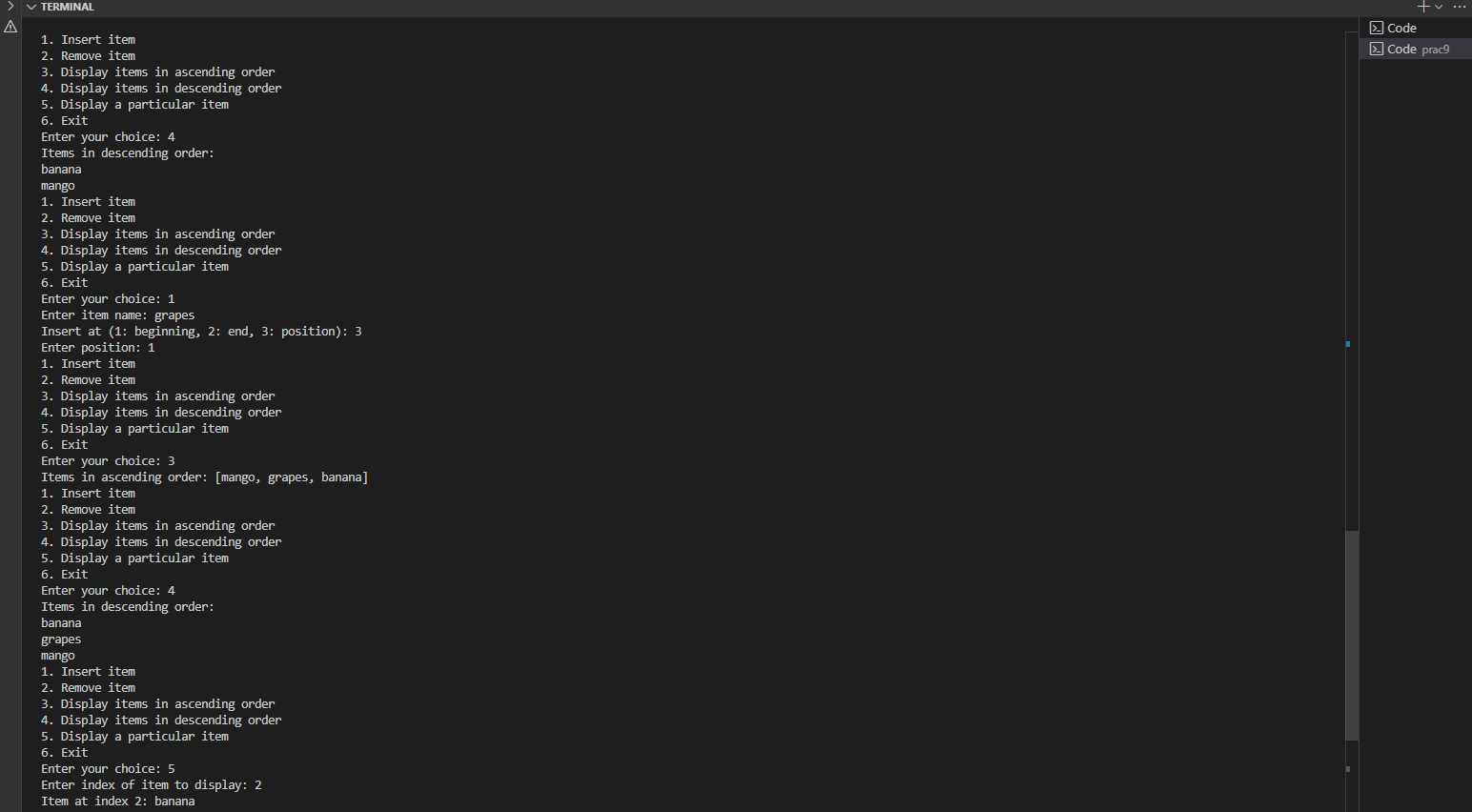
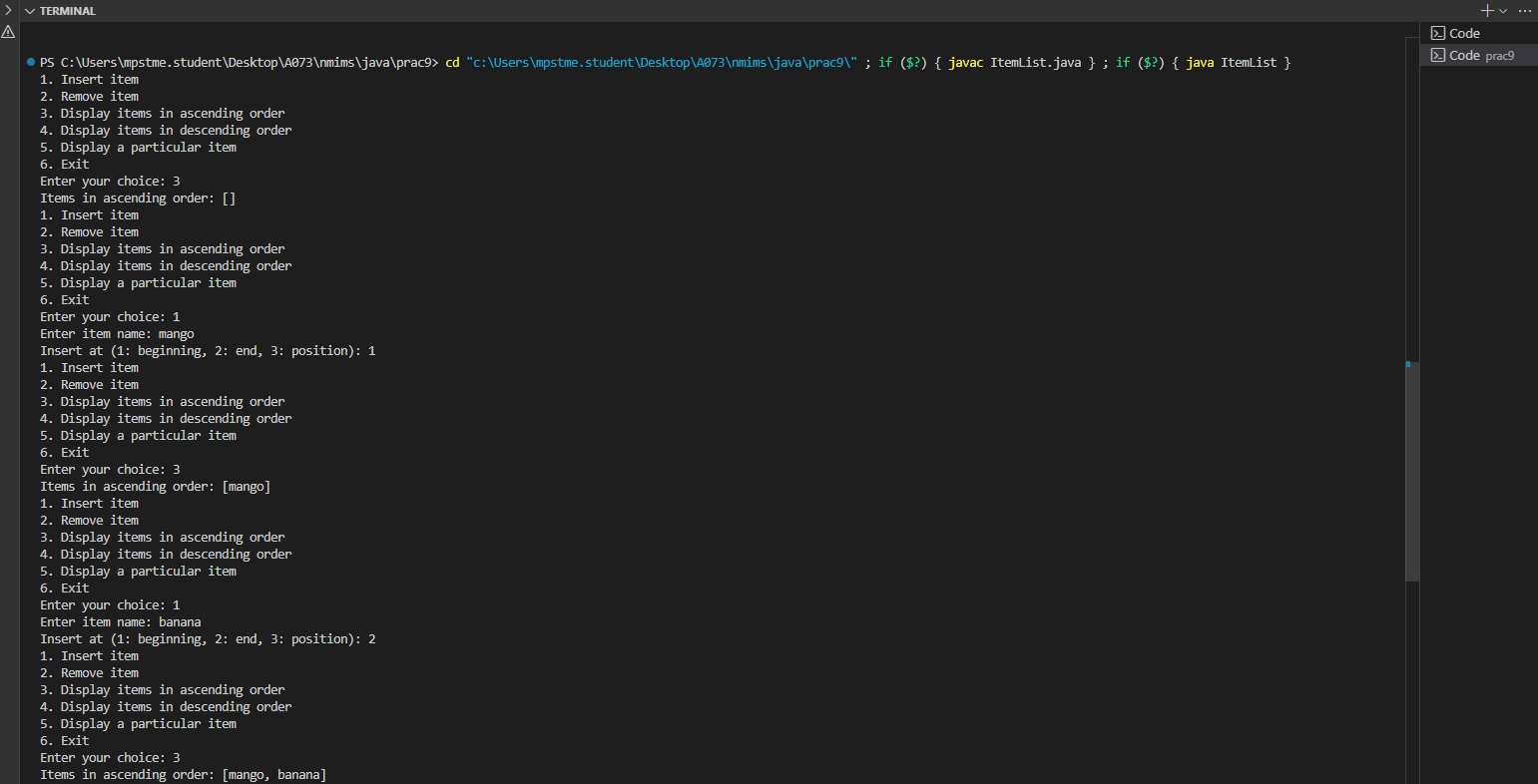
**Task1:**

****

**Task2:**

****

**Task3:**

****

1. **Observation learning and conclusion:** mention what learning you got out of practical

**Successfully implemented and learnt about Java Collections.**

1. **Questions of Curiosity :**
2. Differentiate between iterable interface, iterator interface, iterator method

Iterable Interface: Represents a collection that can be iterated over. It has one method, iterator(), which returns an Iterator.

Iterator Interface: Provides methods to traverse a collection, including hasNext(), next(), and remove().

Iterator Method: Refers to the iterator() method from the Iterable interface that returns an Iterator for a collection.

1. Explain synchronized and non synchronized collections in java.

Synchronized Collections: Thread-safe collections that prevent concurrent modification, wrapped using methods like Collections.synchronizedList().

Non-Synchronized Collections: Standard collections that are not thread-safe and may lead to unpredictable behavior when accessed by multiple threads simultaneously.

1. Why are collections essential in Java programming?

Collections are essential for storing, managing, and manipulating groups of objects efficiently. They provide various data structures and algorithms, enhancing code reusability, maintainability, and performance in managing dynamic data.