**Q.1) Implementation of sorting techniques.**

1. **Insertion Sort**

import java.util.\*;

class SortEx {

public int arr[];

public void insertion\_sort() {

int n = arr.length;

for (int i = 1; i < n; i++) {

int key = arr[i];

int j = i - 1;

while (j >= 0 && arr[j] > key) {

arr[j + 1] = arr[j];

j--;

}

arr[j + 1] = key;

}

public void takeInput() {

Scanner sc = new Scanner(System.in);

System.out.println("Enter size of the array:");

int size = sc.nextInt();

arr = new int[size];

System.out.println("Enter " + size + " elements:");

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

arr[i] = sc.nextInt();

}

public void display() {

for (int i : arr) {

System.out.print(i + " ");

}

System.out.println();

}

}

public class Sort {

public static void main(String[] args) {

SortEx s = new SortEx();

s.takeInput();

System.out.println("Before sort");

s.display();

System.out.println("After sort");

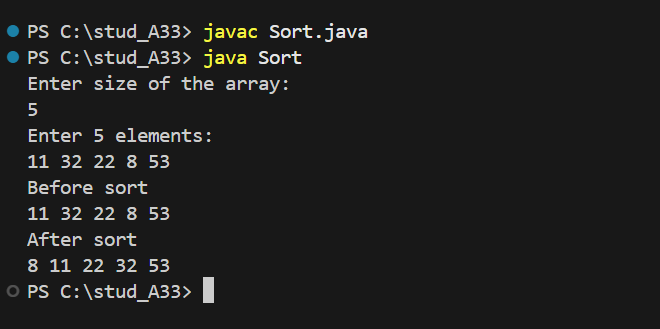
s.insertion\_sort();

s.display();

}

}

Output :

****

1. **Bubble sort**

import java.util.\*;

class SortEx {

public int arr[];

public void bubble\_sort() {

int n = arr.length;

for (int i = 0; i < n; i++) {

boolean swapped = false;

for (int j = 0; j < n - i - 1; j++) {

if (arr[j] > arr[j + 1]) {

int temp = arr[j + 1];

arr[j + 1] = arr[j];

arr[j] = temp;

swapped = true;

}

}

if (!swapped) {

break;

}

}

}

public void takeInput() {

Scanner sc = new Scanner(System.in);

System.out.println("Enter size of the array:");

int size = sc.nextInt();

arr = new int[size];

System.out.println("Enter " + size + " elements:");

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

arr[i] = sc.nextInt();

}

}

public void display() {

for (int i : arr) {

System.out.print(i + " ");

}

System.out.println();

}

}

public class Sort {

public static void main(String[] args) {

SortEx s = new SortEx();

s.takeInput();

System.out.println("Before sort");

s.display();

System.out.println("After sort");

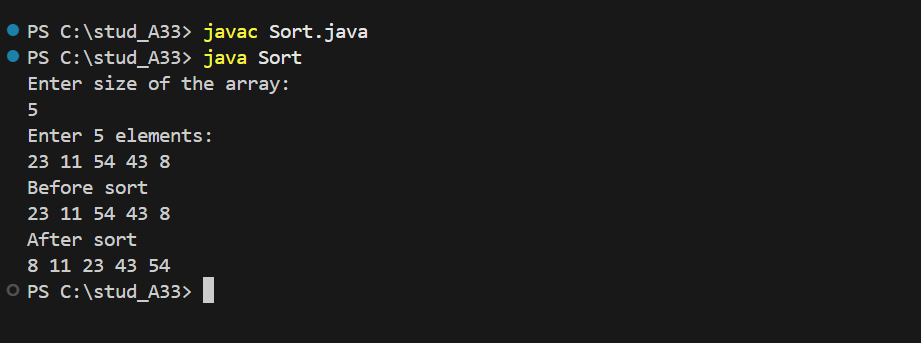
s.bubble\_sort();

s.display();

}

}

Output :



1. **Selection sort**

import java.util.\*;

class SortEx {

public int arr[];

public void selection\_sort() {

int n = arr.length;

for (int i = 0; i < n - 1; i++) {

int minInd = i;

for (int j = i + 1; j < n; j++) {

if (arr[j] < arr[minInd]) {

minInd = j;

}

}

int temp = arr[minInd];

arr[minInd] = arr[i];

arr[i] = temp;

}

}

public void takeInput() {

Scanner sc = new Scanner(System.in);

System.out.println("Enter size of the array:");

int size = sc.nextInt();

arr = new int[size];

System.out.println("Enter " + size + " elements:");

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

arr[i] = sc.nextInt();

}

}

public void display() {

for (int i : arr) {

System.out.print(i + " ");

}

System.out.println();

}

}

public class Sort {

public static void main(String[] args) {

SortEx s = new SortEx();

s.takeInput();

System.out.println("Before sort");

s.display();

System.out.println("After sort");

s.selection\_sort();

s.display();

}

}

Output :



1. **Shell sort**

import java.util.\*;

class SortEx {

public int arr[];

public void shell\_Sort() {

int n = arr.length;

for (int gap = n / 2; gap >= 1; gap = gap / 2) {

for (int j = gap; j < n; j++) {

int temp = arr[j];

int i = j;

while (i >= gap && arr[i - gap] > temp) {

arr[i] = arr[i - gap];

i -= gap;

}

arr[i] = temp;

}

}

}

public void takeInput() {

Scanner sc = new Scanner(System.in);

System.out.println("Enter size of the array:");

int size = sc.nextInt();

arr = new int[size];

System.out.println("Enter " + size + " elements:");

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

arr[i] = sc.nextInt();

}

}

public void display() {

for (int i : arr) {

System.out.print(i + " ");

}

System.out.println();

}

}

public class Sort {

public static void main(String[] args) {

SortEx s = new SortEx();

s.takeInput();

System.out.println("Before sort");

s.display();

System.out.println("After sort");

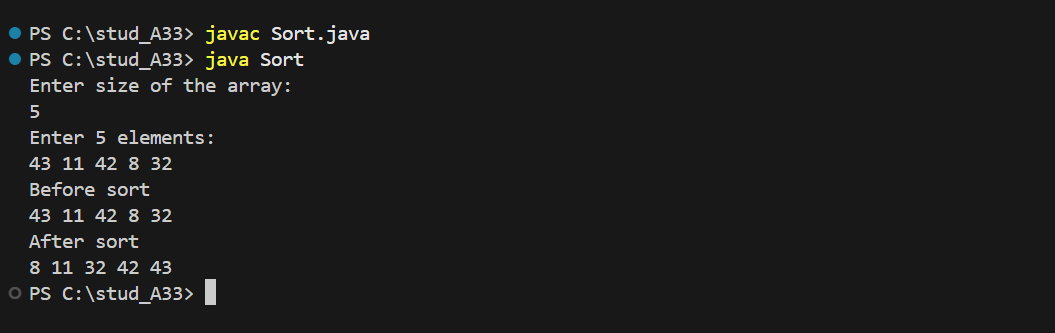
s.shell\_Sort();

s.display();

}

}

Output :



**Q.2) Implementaion of linear and binary search.**

import java.util.Scanner;

class Search {

    public int arr[];

    public boolean linearSearch(int target) {

        for (int i = 0; i < arr.length; i++) {

            if (arr[i] == target) {

                return true;

            }

        }

        return false;

    }

    public void insertion\_sort() {

        int n = arr.length;

        for (int i = 1; i < n; i++) {

            int key = arr[i];

            int j = i - 1;

            while (j >= 0 && arr[j] > key) {

                arr[j + 1] = arr[j];

                j--;

            }

            arr[j + 1] = key;

        }

    }

    public boolean binarySearch(int target) {

        insertion\_sort();

        int start = 0;

        int end = arr.length - 1;

        while (start <= end) {

            int mid = (start + end) / 2;

            if (arr[mid] == target) {

                return true;

            } else if (arr[mid] < target) {

                start = mid + 1;

            } else {

                end = mid - 1;

            }

        }

        return false;

    }

    public void takeInput() {

        Scanner sc = new Scanner(System.in);

        System.out.println("Enter size of the array:");

        int size = sc.nextInt();

        arr = new int[size];

        System.out.println("Enter " + size + " elements :");

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

            arr[i] = sc.nextInt();

        }

    }

    public void display() {

        for (int i : arr) {

            System.out.print(i + " ");

        }

        System.out.println();

    }

}

public class searching {

    public static void main(String[] args) {

        Scanner sc = new Scanner(System.in);

        int option, n;

        boolean ans;

        Search s = new Search();

        s.takeInput();

        char ch = 'Y';

        while (ch == 'Y' || ch == 'y') {

            System.out.println("1.For linear search\n2.For Binary Search");

            option = sc.nextInt();

            switch (option) {

                case 1:

                    System.out.println("Enter target");

                    n = sc.nextInt();

                    ans = s.linearSearch(n);

                    if (ans == true) {

                        System.out.println("target is present");

                    } else {

                        System.out.println("target is not present");

                    }

                    break;

                case 2:

                    System.out.println("Enter target");

                    n = sc.nextInt();

                    ans = s.binarySearch(n);

                    if (ans == true) {

                        System.out.println("target is present");

                    } else {

                        System.out.println("target is not present");

                    }

                    break;

                default:

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

                    break;

            }

            System.out.println("Do yo want to test other searching technique?");

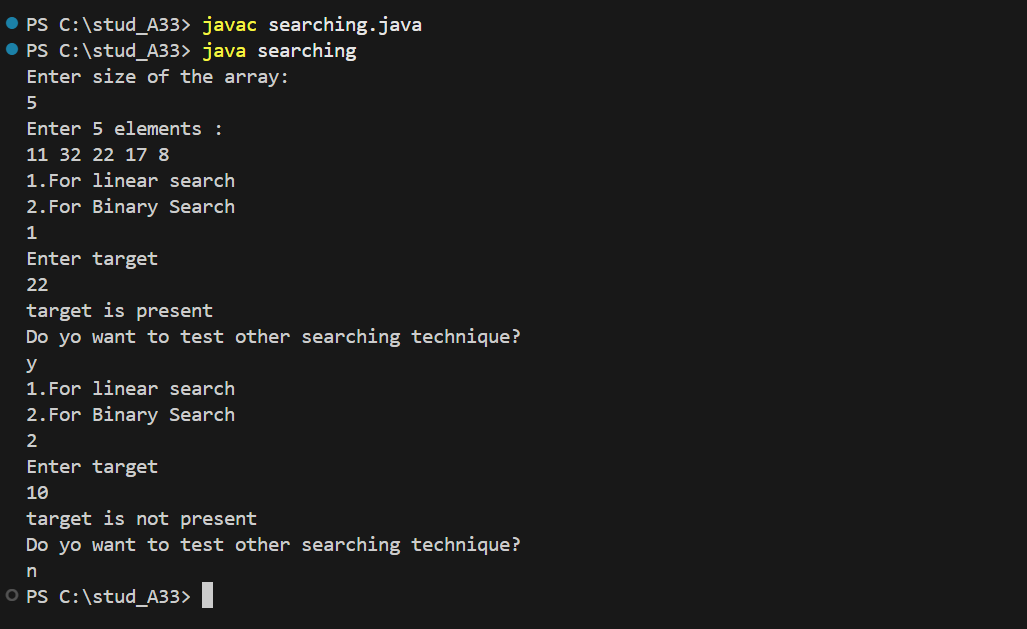
            ch = sc.next().charAt(0);

        }

    }

}

**Output :**

****

**Q.3) Method for hashing : Modulo division, Digit Extraction and Linear probe for collision resolution.**

import java.util.Scanner;

class HashEx {

    private long[] arr = new long[20];

    public HashEx() {

        for (int i = 0; i < 20; i++) {

            arr[i] = 0;

        }

    }

    void directMethod() {

        Scanner sc = new Scanner(System.in);

        System.out.println("Enter key");

        int key = sc.nextInt();

        if (key >= 1 && key < 20) {

            arr[key] = key;

            System.out.println("Key entered");

        } else {

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

        }

    }

    void subtractionMethod() {

        Scanner sc = new Scanner(System.in);

        System.out.println("Enter key");

        int key = sc.nextInt();

        if (key >= 81 && key < 99) {

            arr[100 - key] = key;

            System.out.println("Key entered");

        } else {

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

        }

    }

    void ModuloDivisionMethod() {

        Scanner sc = new Scanner(System.in);

        System.out.println("Enter key");

        int key = sc.nextInt();

        arr[(key % 20) + 1] = key;

        System.out.println("Key entered");

    }

    void ModuloDivisionWithCollision() {

        Scanner sc = new Scanner(System.in);

        System.out.println("Enter key");

        int key = sc.nextInt();

        int address = (key % 20) + 1;

        if (arr[address] == 0) {

            arr[address] = key;

        } else {

            arr[address + 1] = key;

        }

        System.out.println("Key entered");

    }

    void digitExtractionMethod() {

        Scanner sc = new Scanner(System.in);

        System.out.println("Enter four digit key");

        int key = sc.nextInt();

        arr[((key / 10) % 20) + 1] = key;

        System.out.println("Key entered");

    }

    void digitExtractionMethodwithCollision() {

        Scanner sc = new Scanner(System.in);

        System.out.println("Enter four digit key");

        int key = sc.nextInt();

        int address = (((key / 10) + 1) % 20) + 1;

        if (arr[address] == 0) {

            arr[address] = key;

        } else {

            arr[address + 1] = key;

        }

        System.out.println("Key entered");

    }

    void display() {

        for (int i = 0; i < 20; i++) {

            System.out.print(arr[i] + " ");

        }

        System.out.println();

    }

}

public class hashing {

    public static void main(String[] args) {

        HashEx h = new HashEx();

        Scanner sc = new Scanner(System.in);

        char ch = 'y';

        while (ch == 'y') {

            System.out.println("1. Direct Method");

            System.out.println("2. Subtraction Method");

            System.out.println("3. Modulo Division Method");

            System.out.println("4. Modulo Division With Collision");

            System.out.println("5. Digit Extraction Method");

            System.out.println("6. Digit Extraction Method with Collision");

            System.out.println("7. Display");

            System.out.println("Enter an Option");

            int option = sc.nextInt();

            switch (option) {

                case 1 ->

                    h.directMethod();

                case 2 ->

                    h.subtractionMethod();

                case 3 ->

                    h.ModuloDivisionMethod();

                case 4 ->

                    h.ModuloDivisionWithCollision();

                case 5 ->

                    h.digitExtractionMethod();

                case 6 ->

                    h.digitExtractionMethodwithCollision();

                case 7 ->

                    h.display();

            }

            System.out.println("Do you want to continue? (y/n):");

            ch = sc.next().charAt(0);

        }

    }

}

Output :

