**LAB Assignment Submission**

for

**Data Structures and Algorithms**

**Course Code: CSE2711**

**B.Tech CSE-VII/ECOM**

**Batch 2024**

**Name:** Vedansh Mathur

**Enrollment Number:** 240686

**Problem Statement-**

Problem 1: Bubble Sort

Write a program to sort an array of integers using the Bubble Sort algorithm.

 Input: An array of n integers (entered by the user).

 Output: The array sorted in ascending order.

 Additionally, display the number of comparisons and swaps made during sorting.

Example:

Input: 5 2 9 1 5 6

Output: 1 2 5 5 6 9

Comparisons: 15

Swaps: 6

Problem 2: Insertion Sort

Write a program to implement Insertion Sort to arrange a list of student marks in

ascending order.

 Input: A list of marks scored by students in a class test.

 Output: Sorted list of marks.

 Also, print the array after each insertion step to visualize the sorting process.

Example:

Input: 40 20 60 10 50

Step 1: 20 40 60 10 50

Step 2: 20 40 60 10 50

Step 3: 10 20 40 60 50

Step 4: 10 20 40 50 60

Final Output: 10 20 40 50 60

Problem 3: Selection Sort

Write a program to sort an array of names (strings) using the Selection Sort algorithm.

 Input: A list of n names.

 Output: The list of names sorted in alphabetical order.

 Also, display which element was selected as the minimum at each iteration.

Example:

Input: ["John", "Alice", "Bob", "Larry"]

Iteration 1 → Minimum = "Alice" → Swap with "John"

Iteration 2 → Minimum = "Bob" → Swap with "John"

Iteration 3 → Minimum = "John" → Already in place

Output: ["Alice", "Bob", "John", “Larry”]

Problem 4: Validate Stability of Sorting Algorithms

Write a program to verify the Stability of the above sorting algorithms (Bubble/Insertion/

Selection Sort). A sorting algorithm is Stable if it preserves the relative order of elements

with equal keys (duplicate elements).

Input: Your choice (decide the data to be used)

Output: Sorted data (display the βinal output)

**Solution –**

**A1->**

#include <iostream>

using namespace std;

void sorting\_arr(int arr[], int n){

    int comparisons = 0;

    int swaps = 0;

    for (int i =0; i<n-1; i++){

        comparisons++;

        for (int j=0; j<n-i-1; j++){

            if (arr[j] > arr[j+1]){

                swap(arr[j], arr[j+1]);

                swaps++;

            }

        }

    }

    cout<<"The sorted array is: ";

    for (int i =0; i<n; i++){

        cout<<arr[i]<<" "<<endl;

    }

    cout << "Comparisons: " << comparisons << endl;

    cout << "Swaps: " << swaps << endl;

}

int main () {

    int arr[100];

    cout<<"Enter the size of the array: ";

    int n;

    cin>>n;

    cout<<"Enter the elements: ";

    for (int i= 0; i<n; i++){

        cin>>arr[i];

    }

    sorting\_arr (arr, n);

    return 0;

}

**A2->**

#include <iostream>

using namespace std;

void insertion\_sort(int arr[], int n){

    for (int i = 1; i<n; i++){

        int curr = arr[i];

        int prev = i-1;

        while(prev >=0 && curr < arr[prev]){

            arr[prev +1] = arr[prev];

            prev--;

        }

        arr[prev+1] = curr;

        cout << "Step " << i << ": ";

        for (int k = 0; k < n; k++) {

            cout << arr[k] << " ";

        }

        cout << endl;

    }

    cout << "Final Output: ";

    for (int i = 0; i < n; i++) {

        cout << arr[i] << " ";

    }

    cout << endl;

}

int main () {

    int arr[] = {56, 25, 2, 32, 89};

    int n = sizeof(arr) / sizeof(arr[0]);

    insertion\_sort(arr, n);

    return 0;

}

**A3->**

#include <iostream>

#include <string>

using namespace std;

void selection\_sort(string arr[], int n) {

    for (int i = 0; i < n - 1; i++) {

        int minIndex = i;

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

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

                minIndex = j;

            }

        }

        cout << "Iteration " << i + 1 << " -> Minimum = \"" << arr[minIndex] << "\"";

        if (minIndex != i) {

            swap(arr[i], arr[minIndex]);

            cout << " -> Swap with \"" << arr[i] << "\"";

        } else {

            cout << " -> Already in place";

        }

        cout << endl;

    }

    cout << "Final Output: ";

    for (int i = 0; i < n; i++) {

        cout << arr[i] << " ";

    }

}

int main() {

    int n;

    cout << "Enter number of names: ";

    cin >> n;

    string arr[100];

    cout << "Enter names: ";

    for (int i = 0; i < n; i++) {

        cin >> arr[i];

    }

    selection\_sort(arr, n);

    return 0;

}

**A4->**

#include <iostream>

#include <string>

using namespace std;

struct Student {

    int marks;

    string name;

};

// Bubble Sort (Stable)

void bubble\_sort(Student arr[], int n) {

    for (int i=0; i<n-1; i++) {

        for (int j=0; j<n-i-1; j++) {

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

                swap(arr[j], arr[j+1]);

            }

        }

    }

    cout << "Bubble Sort (Stable) -> ";

    for (int i=0; i<n; i++) {

        cout << "(" << arr[i].marks << "," << arr[i].name << ") ";

    }

    cout << endl;

}

// Insertion Sort (Stable)

void insertion\_sort(Student arr[], int n) {

    for (int i=1; i<n; i++) {

        Student curr = arr[i];

        int prev = i-1;

        while (prev >=0 && arr[prev].marks > curr.marks) {

            arr[prev+1] = arr[prev];

            prev--;

        }

        arr[prev+1] = curr;

    }

    cout << "Insertion Sort (Stable) -> ";

    for (int i=0; i<n; i++) {

        cout << "(" << arr[i].marks << "," << arr[i].name << ") ";

    }

    cout << endl;

}

// Selection Sort (NOT Stable)

void selection\_sort(Student arr[], int n) {

    for (int i=0; i<n-1; i++) {

        int minIndex = i;

        for (int j=i+1; j<n; j++) {

            if (arr[j].marks < arr[minIndex].marks) {

                minIndex = j;

            }

        }

        swap(arr[i], arr[minIndex]);

    }

    cout << "Selection Sort (Not Stable) -> ";

    for (int i=0; i<n; i++) {

        cout << "(" << arr[i].marks << "," << arr[i].name << ") ";

    }

    cout << endl;

}

int main() {

    int n = 5;

    Student arr[5] = {

        {50, "A"}, {40, "B"}, {50, "C"}, {30, "D"}, {40, "E"}

    };

    Student b[5], ins[5], sel[5];

    for (int i=0; i<n; i++) {

        b[i]=arr[i];

        ins[i]=arr[i];

        sel[i]=arr[i];

    }

    bubble\_sort(b, n);

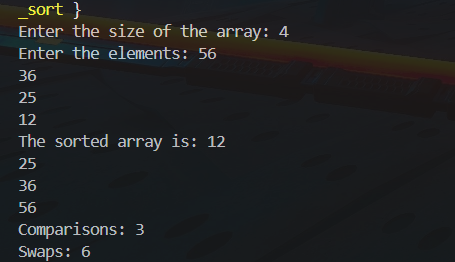
    insertion\_sort(ins, n);

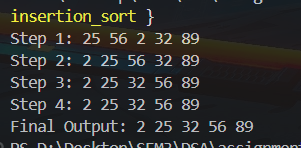
    selection\_sort(sel, n);

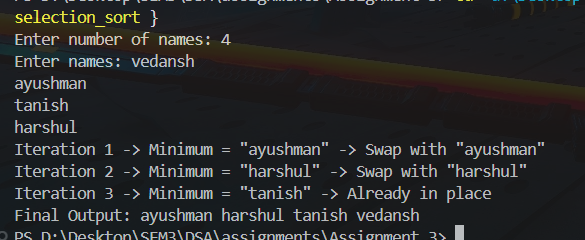
    return 0;

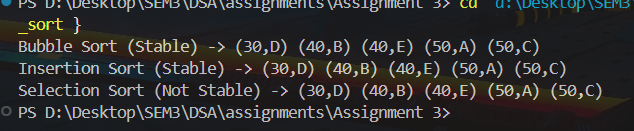
}

**Outputs-**

**

**

**

**