**DSA LAB – 9**

**Name:** Etcherla Sai Manoj **Mis. No:** 112015044 **Branch:** CSE

**Question 1:**

**Code:**

#include<iostream>

using namespace std;

int Linear\_search(int array[], int n, int x){

cout << "Implementing Linear Search...\n";

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

if(array[i] == x){

cout << "The element " << x << " is found at position " << i << " in the array\n";

return 0;

}

}

cout << "The element is not present in the array\n";

return 0;

}

int Binary\_search(int array[], int left, int right, int x){

cout << "Implementing Binary Search...\n";

while(left <= right){

int middle = left + (right - left) / 2;

if(array[middle] == x){

cout << "The element " << x << " is found at position " << middle << " in the array\n";

return 0;

}

if(array[middle] > x){

right = middle - 1;

}

else{

left = middle + 1;

}

}

cout << "The element is not present in the array\n";

return 0;

}

void Bubble\_sort(int array[], int n){

cout << "Implementing Bubble Sort...\n";

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

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

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

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

}

}

}

}

void Insertion\_sort(int array[], int n){

cout << "Implementing Insertion Sort...\n";

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

int temp = array[i];

int j = i -1;

while(temp <= array[j] && j >= 0){

array[j+1] = array[j];

j = j - 1;

}

array[j+1] = temp;

}

}

void Selection\_sort(int array[], int n){

cout << "Implementing Selection Sort...\n";

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

int min\_idx = i;

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

if (array[j] < array[min\_idx]){

min\_idx = j;

}

swap(array[min\_idx], array[i]);

}

}

void Merge(int array[], int start, int middle, int end){

int index = 0, temp[end - start + 1];

int i = start, j = middle + 1;

while(i <= middle && j <= end){

if(array[i] <= array[j]){

temp[index] = array[i];

i = i + 1;

}

else{

temp[index] = array[j];

j = j + 1;

}

index = index + 1;

}

while(i <= middle){

temp[index] = array[i];

index = index + 1;

i = i + 1;

}

while(j <= end){

temp[index] = array[j];

index= index + 1;

j = j + 1;

}

for(i = start; i <= end; i++){

array[i] = temp[i - start];

}

}

void Merge\_sort(int array[], int start, int end){

if(start < end){

int middle = (start + end) / 2;

Merge\_sort(array, start, middle);

Merge\_sort(array, middle + 1, end);

Merge(array, start, middle, end);

}

}

void heapify(int array[], int n, int i){

int largest = i;

int left = 2 \* i + 1;

int right = 2 \* i + 2;

if(left < n && array[left] > array[largest]){

largest = left;

}

if(right < n && array[right] > array[largest]){

largest = right;

}

if(largest != i){

swap(array[i], array[largest]);

heapify(array, n, largest);

}

}

void Heap\_sort(int array[], int n){

cout << "Implementing Heap Sort...\n";

for(int i = n / 2; i >= 0; i--){

heapify(array, n, i);

}

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

swap(array[0], array[i]);

heapify(array, i, 0);

}

}

int partition(int array[], int start, int end){

int left = start, right = end, loc = start, flag = 0;

while(flag == 0){

while(array[loc] <= array[right] && loc != right){

right = right - 1;

}

if(loc == right){

flag = 1;

}

else if(array[loc] > array[right]){

swap(array[loc], array[right]);

loc = right;

}

if(flag == 0){

while(array[loc] >= array[left] && loc != left){

left = left + 1;

}

if(loc == left){

flag = 1;

}

else if(array[loc] < array[left]){

swap(array[loc], array[left]);

loc = left;

}

}

}

return loc;

}

void Quick\_sort(int array[], int start, int end){

int loc;

if(start < end){

loc = partition(array, start, end);

Quick\_sort(array, start, loc - 1);

Quick\_sort(array, loc + 1, end);

}

}

void display(int array[], int n){

cout << "Sorted array : ";

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

cout << array[i] << " ";

}

cout << "\n";

}

int main(){

int n, choice;

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

cin >> n;

int array[n];

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

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

cin >> array[i];

}

cout << "===================Menu===================\n";

cout << "1. Linear Search\n";

cout << "2. Binary Search\n";

cout << "3. Bubble Sort\n";

cout << "4. Insertion Sort\n";

cout << "5. Selection Sort\n";

cout << "6. Merge Sort\n";

cout << "7. Heap Sort\n";

cout << "8. Quick Sort\n";

cout << "9. EXIT\n";

cout << "==========================================\n";

while(1){

cout << "\nEnter your choice to perform : ";

cin >> choice;

switch(choice)

{

case 1:

int a;

cout << "Enter the value you want to search : ";

cin >> a;

Linear\_search(array, n, a);

break;

case 2:

int b;

cout << "Enter the value you want to search : ";

cin >> b;

Binary\_search(array, 0, n-1, b);

break;

case 3:

Bubble\_sort(array, n);

display(array, n);

break;

case 4:

Insertion\_sort(array, n);

display(array, n);

break;

case 5:

Selection\_sort(array, n);

display(array, n);

break;

case 6:

cout << "Implementing Merge Sort...\n";

Merge\_sort(array, 0, n -1);

display(array, n);

break;

case 7:

Heap\_sort(array, n);

display(array, n);

break;

case 8:

cout << "Implementing Quick Sort...\n";

Quick\_sort(array, 0, n-1);

display(array, n);

break;

case 9:

return 0;

default:

cout << "Enter valid choice...!!!\n";

break;

}

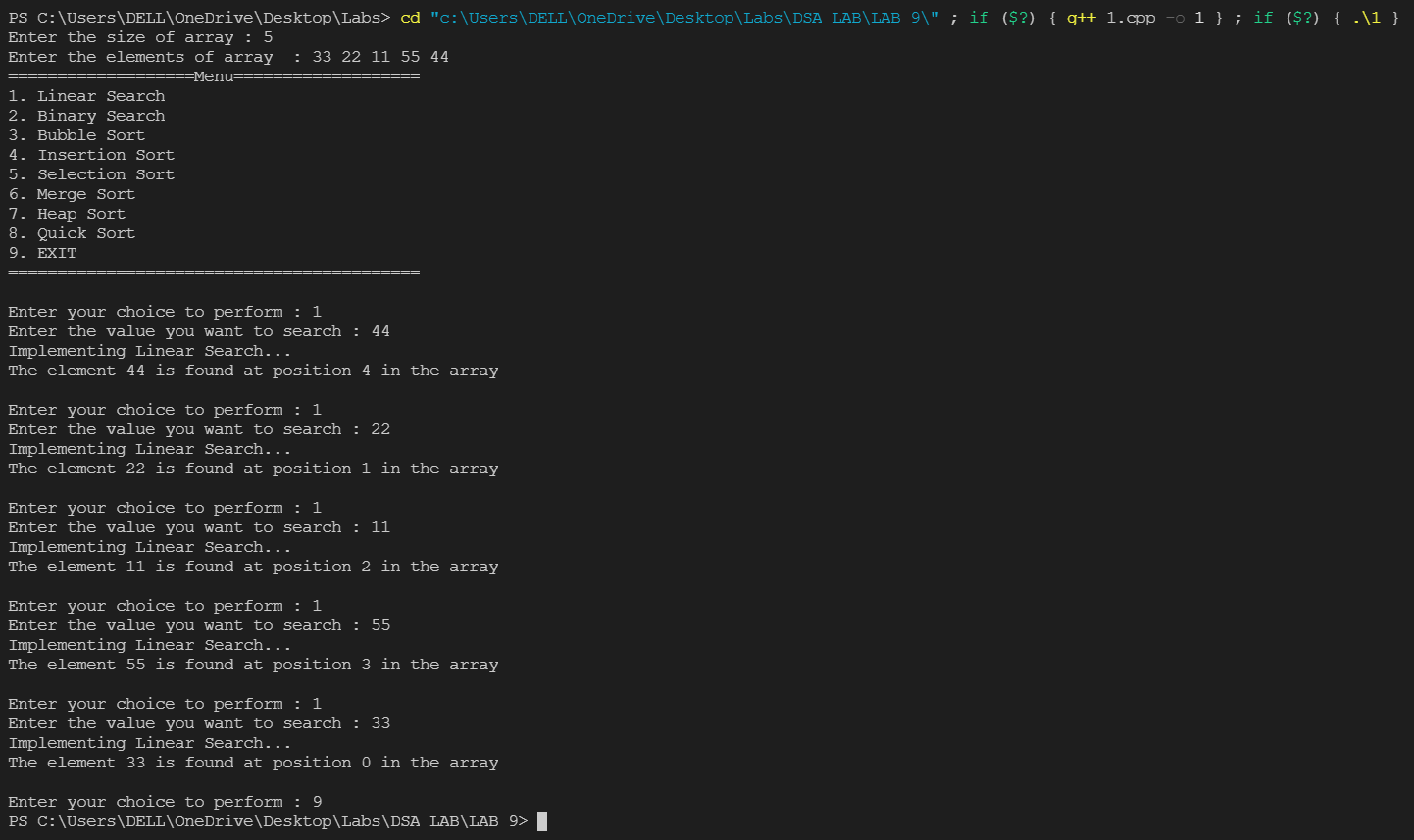
}

return 0;

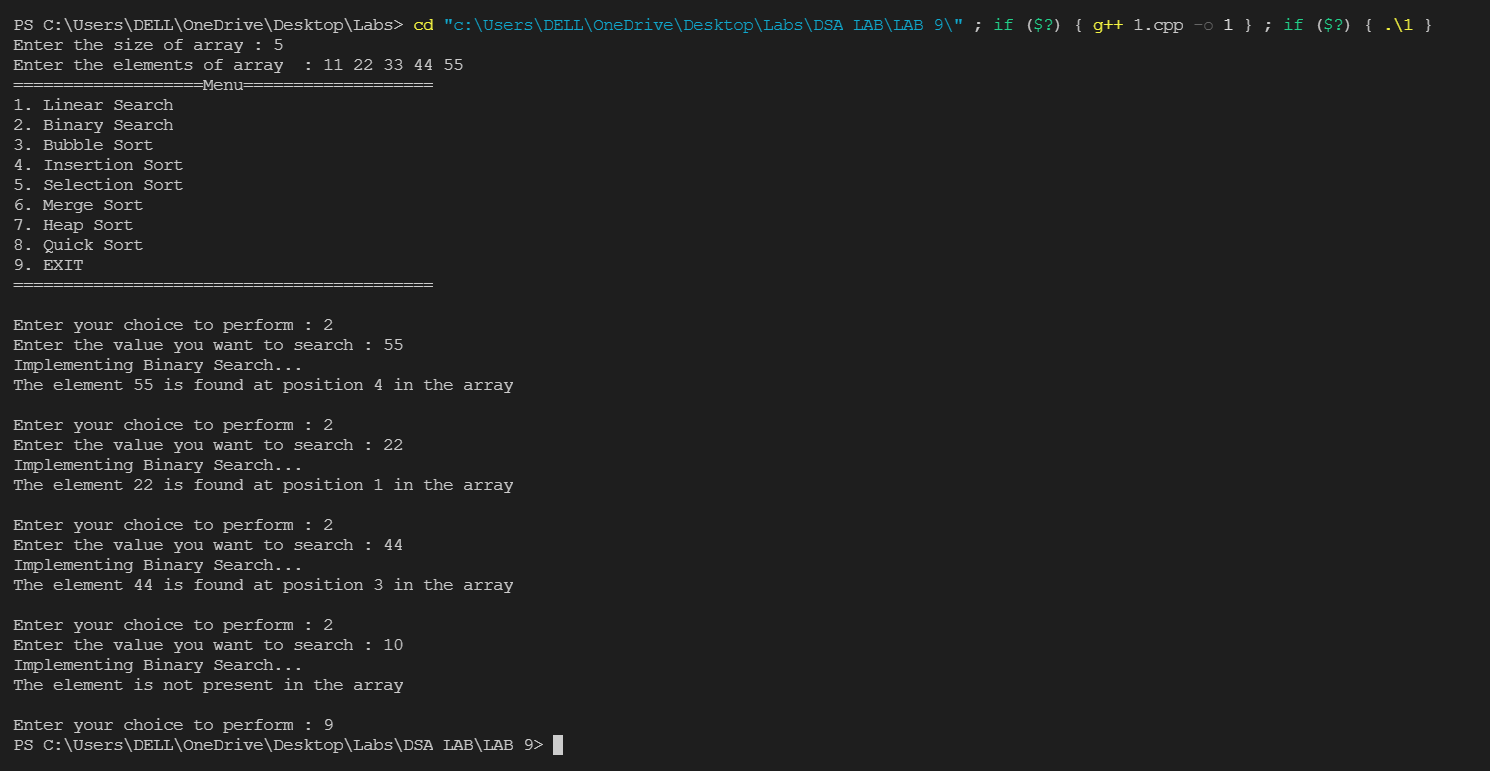
}

**Input & Output:**

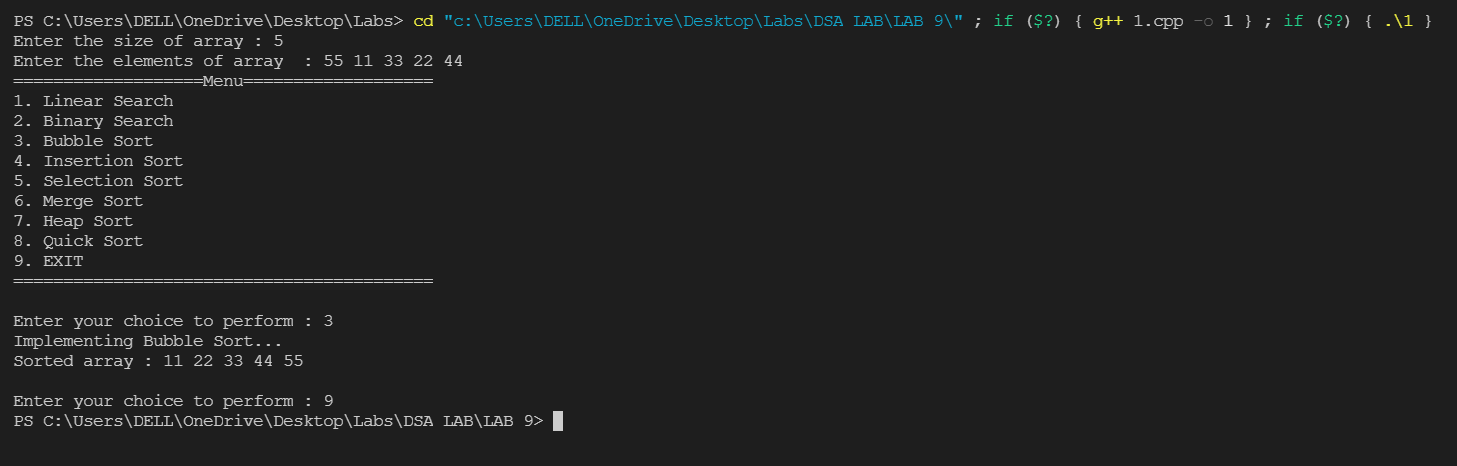
Linear Search:



Binary Search:



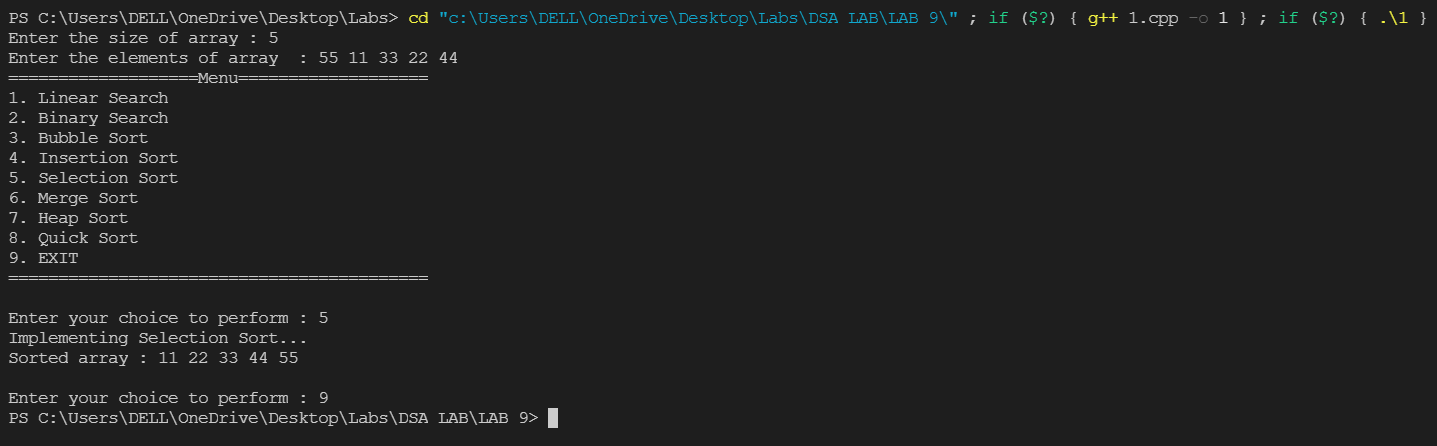
Bubble Sort:



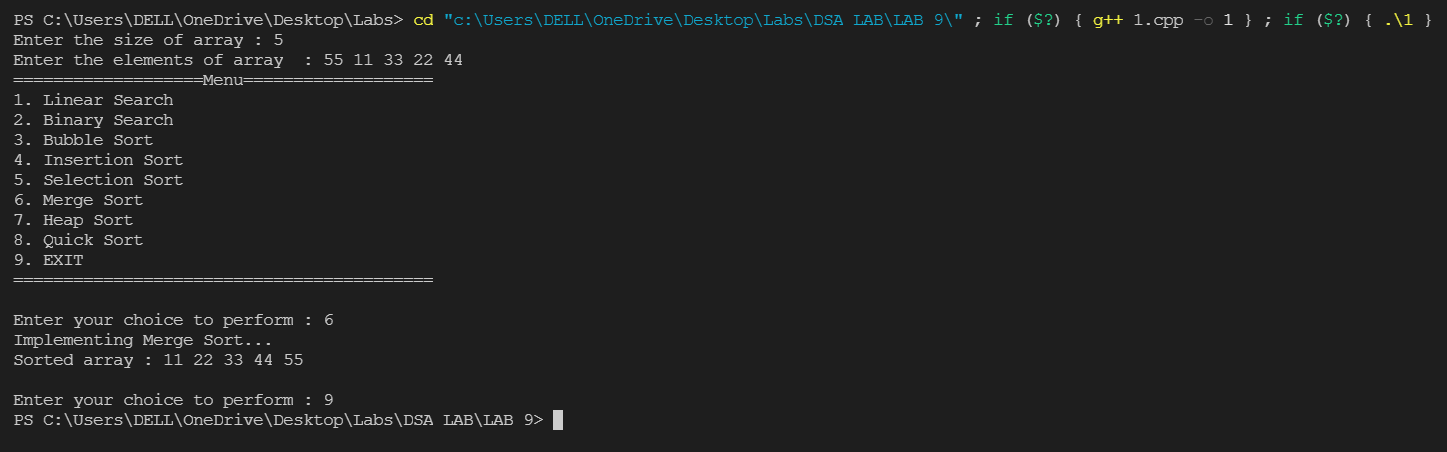
Insertion Sort:



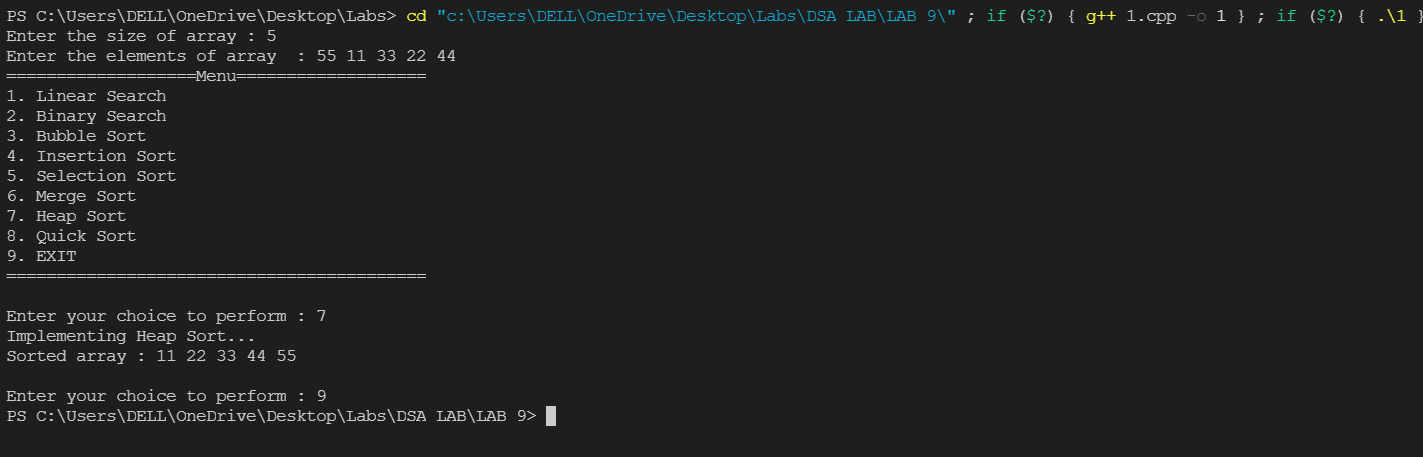
Selection Sort:



Merge Sort:



Heap Sort:



Quick Sort:

