Experiment Number: 7

Title: Implementation of Bubble Sort.

Problem Statement: Write a C++ program to arrange the given set of numbers in ascending order using Bubble sort.

Algorithm:

1. Create the array of size n which is given by the user.
2. Read the array elements from the user.
3. Take the array element and compare it with the next consecutive array element.
4. If the array element is greater than the next consecutive element then swap the elements and repeat the step 3 by taking next element until all the array elements are processed.
5. If the array element is not greater than the next consecutive element then do nothing and repeat the step 3 by taking next element until all the array elements are over.
6. Repeat the step 3 n-1 times which is the number of passes required for bubble sort.

Code:

// Bubble Sort

// To sort given elements in Ascending order

#include<iostream>

using namespace std;

int main()

{

int array[50], n, i, j, k, temp;

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

cin>>n;

cout<<"Enter the array elements to be sorted: ";

//To read the array

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

cin>>array[i];

cout<<"Array before sorting :";

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

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

for(i=1;i<n;++i) // i keeps track of the no. of passes

{

{

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

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

{

temp=array[j];

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

array[j+1]=temp;

}

}

//To print the array elements after every pass

cout<<"\n\n Array after Pass "<<i<<":" ;

for(k=0;k<n;++k)

cout<<" "<<array[k];

}

cout<<"\n\nArray after bubble sort:";

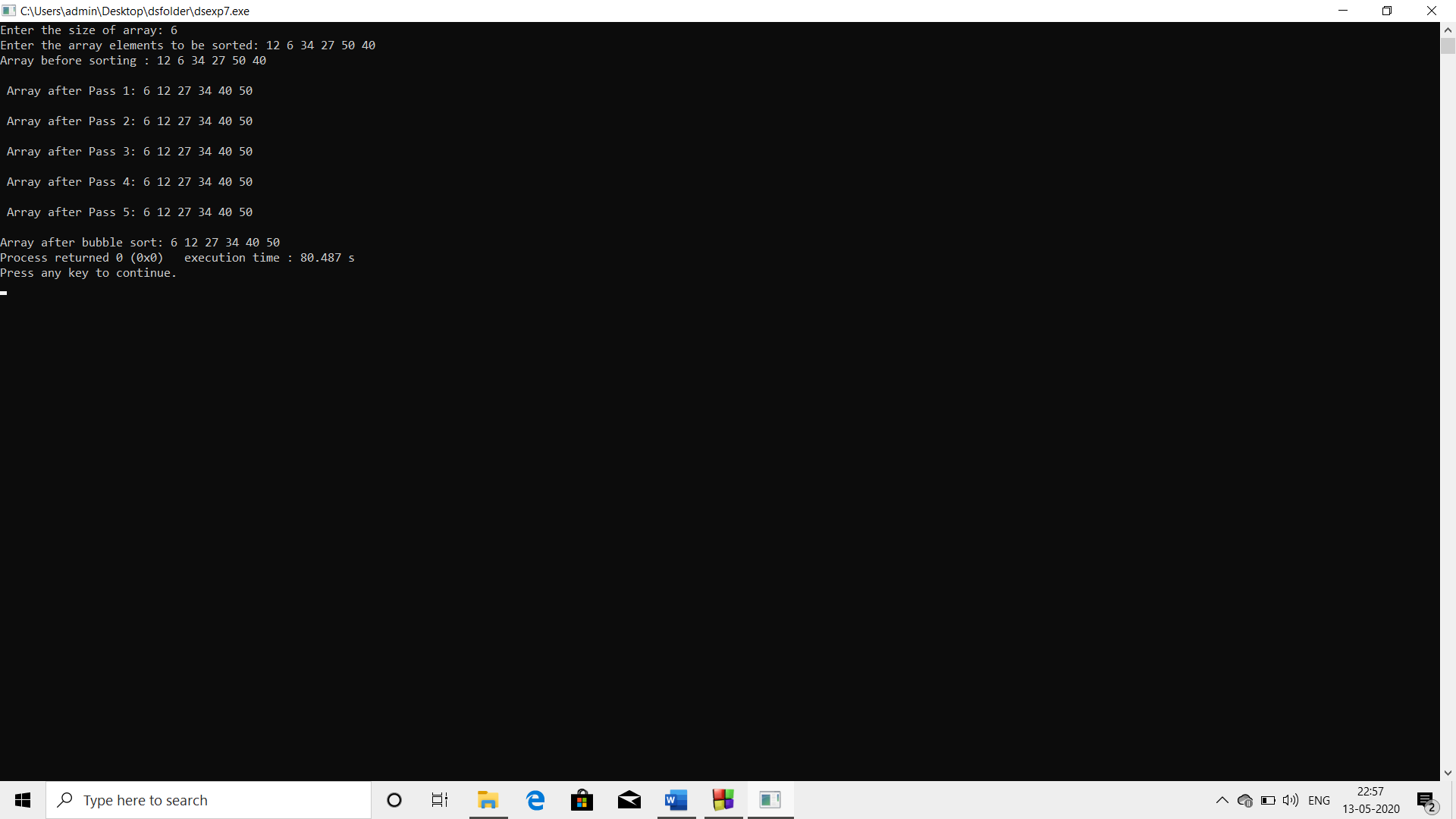
for(i=0;i<n;++i)

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

return 0;

}

Sample Input/ Output:



Analysis:

* Bubble sort is the easiest to understand. However, it is the least efficient. Bubble Sort is the simplest sorting algorithm that works by repeatedly swapping the adjacent elements if they are in wrong order. Given a list of n elements, bubble sort requires (n-1) passes or iterations to sort the data. Therefore bubble sort is more time consuming operation.