Heaven’s light is our guide



**Rajshahi University of Engineering and Technology**

**Dept.: Electrical & Computer Engineering**

**Lab Report- 1**

**Course title : Data Structure & Algorithms Sessional**

**Course code : ECE- 2104**

**Date of exp. : 01-12-2024**

**Date of submission : 07-12-2024**

|  |  |
| --- | --- |
| **Submitted to:** | **Submitted by:** |
| **Md. Omaer Faruq Goni**  Assistant Professor  Electrical & Computer Engineering  RUET | **Asraful Islam**  ID : 2210016  Reg. No. : 1070  Session : 2022-2023  Dept. of ECE, RUET |

**Problem- 1 : Traversing of an array.**

**Code :**

#include<iostream>

using namespace std;

int main (){

int a[100];

int n;

cout<<"Enter number of how many arrays you want to store : ";

cin>>n;

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

cin>>a[i];

}

cout<<"Traversing result : "<<endl;

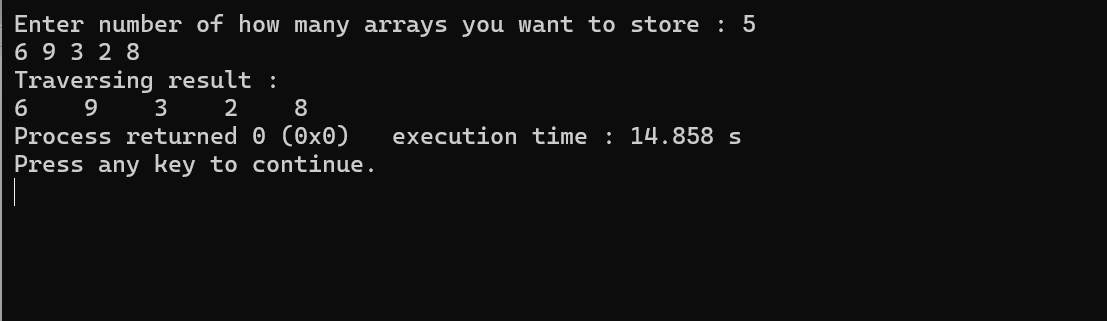
for(int i=0; i<n; i++){

cout<<a[i]<<" ";

}

}

**Output:**



**Discussion :**

Five array elements are stored in this code. The code resolves the traversal problem since each element's access to the output screen shows the traversing process.

**Problem- 2 : Insertation of an element in an array.**

**Code :**

#include<iostream>

using namespace std;

int main (){

int a[100];

int n,p;

cout<<"Enter number of how many arrays you want to store : ";

cin>>n;

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

cin>>a[i];

}

cout<<"Stored array elements : "<<endl;

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

cout<<a[i]<<" ";

}

cout<<endl;

cout<<"Enter position to insert an array : ";

cin>>p;

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

a[i]=a[i-1];

}

cout<<"Enter the new array element : ";

cin>>a[p-1];

cout<<"Result after insertation : "<<endl;

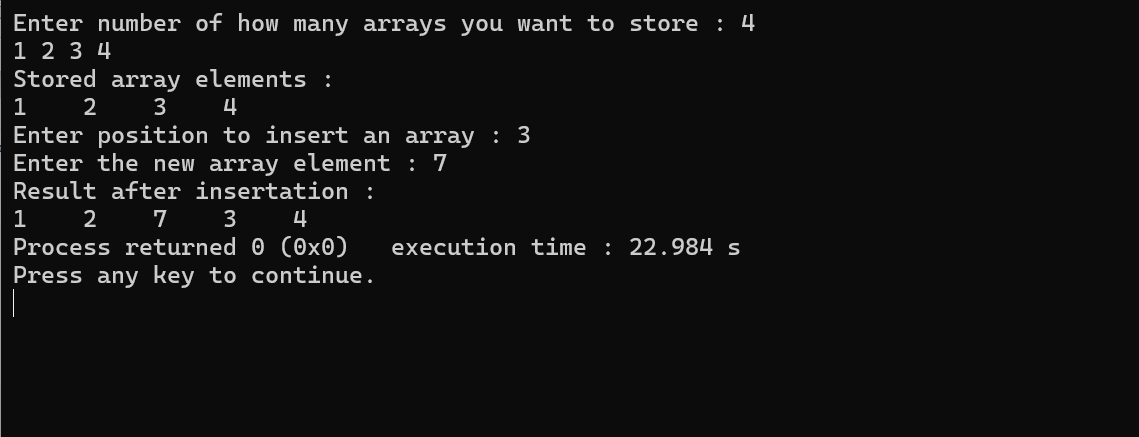
for(int i=0; i<=n; i++){

cout<<a[i]<<" ";

}

}

**Output:**



**Discussion :**

The total array memory in this code is 4. A new array element must be started at the third position. The array's total memory grows by one position as a result. Additionally, the new element is kept in the ordered place. When a new element is inserted, every element in the ordered location moves by one place. Seven is kept here, at the third place. As a result, everything from the third location moves to the right by one place. The addition of an element to the array is thus satisfied.

**Problem- 3 : Deletation of an element from an array.**

**Code :**

#include<iostream>

using namespace std;

int main (){

int a[100];

int n,p;

cout<<"Enter number of how many arrays you want to store : ";

cin>>n;

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

cin>>a[i];

}

cout<<"Stored array elements : "<<endl;

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

cout<<a[i]<<" ";

}

cout<<endl;

cout<<"Enter position to delete it's element : ";

cin>>p;

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

a[i]=a[i+1];

}

cout<<"Result after deletion : "<<endl;

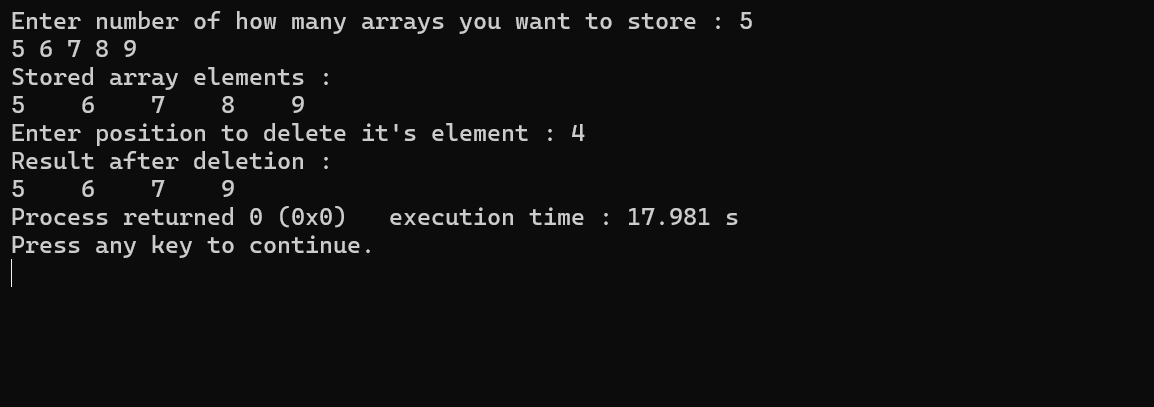
for(int i=0; i<n-1; i++){

cout<<a[i]<<" ";

}

}

**Output :**



**Discussion :**

In this code, forth element 8 is deleted by command. So, at the result, the commanded element is deleted and the total memory decreases by one location.

**Problem- 4 : Bubble sorting of an array.**

**Code :**

#include<iostream>

using namespace std;

int main (){

int a[100];

int n,p;

cout<<"Enter number of how many arrays you want to store : ";

cin>>n;

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

cin>>a[i];

}

cout<<"Stored array elements : "<<endl;

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

cout<<a[i]<<" ";

}

cout<<endl;

cout<<"Process of sorting stored elements : "<<endl;

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

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

if(a[i]<a[i+1]) continue;

else {

int t;

t=a[i];

a[i]=a[i+1];

a[i+1]=t;

}

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

cout<<a[i]<<" ";

}

cout<<endl;

}

}

cout<<endl;

cout<<"Result after sorting : "<<endl;

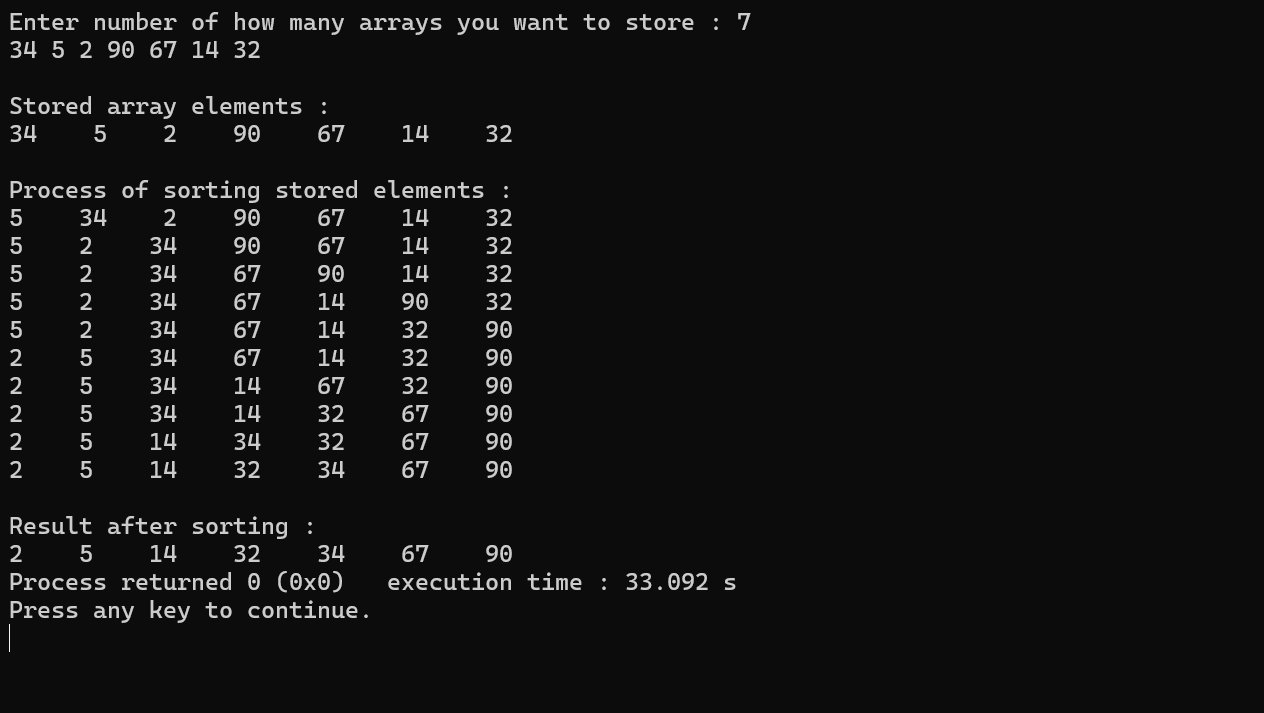
for(int i=0; i<n; i++){

cout<<a[i]<<" ";

}

}

**Output :**



**Discussion :**

In this code, bubble sorting is utilized to arrange a discontinuous array provided as input. Through the bubble sorting process, all elements are accessed sequentially. Two loops are employed to organize the elements in ascending order. After completing the bubble sorting process, the elements are displayed in ascending order as the final output. This procedure ensures the bubble sorting property is maintained.

**==0==**