Arrays

An array is a series of elements of the same type placed in contiguous memory locations that can be individually referenced by adding an index to a unique identifier. It can also be defined as a variable of elements with same datatype meaning we can't have an array that contains strings and characters or whatever data type you may think of.

Array declaration by specifying size

An array can be declared as follows:

```
int main(){
  int my_array [5];
   return 0;
}
```

my_array will be a variable of 5 integers with indexes 0 to 4, meaning when indexing an element that is in this array the index starts from 0 to 4, 0 being the first index of an element in that array and 1 being the second all the way to 4 being the element number 5.

Array declaration by initializing elements.

An array can be declared and being initialised as follows:

```
int main(){
    char my_array []={'1','a','r','6'};
    return 0;
}
```

we declared an array my_array of characters and initialises its values to be `{'1','a','r','6'};`

`the element on index 0 is character 1` *-which is the first character*

`the element on index 1 is character a` *-which is the character*

all the way to the last character 6.

NB: ***Array elements are accessed by using an integer index. Array index starts with 0 and goes till size of array minus 1***

Array declaration by specifying size and initializing elements

Arrays can be declared, initialised and have a specified size this can be done as follow.

```
int main(){
    int arr[6] = { 10, 20, 30, 40 };
    return 0;
}
```

In our case we said, "Hey C++ we want to create an array of integers this array is of size 6 but have 4 values. " In this case C++ will fill the remaining two values with zeros. Let's print all the values that are in that array and see what the output will be.

```
#include <iostream>
#include <vector>
using namespace std;
//#include "sqlite/sqlite3.h"
int main(){
    int arr[6] = { 10, 20, 30, 40 };
```

```
    for(int i=0;i<6;i++){
        cout<<arr[i]<<endl;
    }
    return 0;
}
```

Output:

```
10
20
30
40
0
0

Process finished with exit code 0
```

Finding the size of an array:

To find the size of an array is one of the easiest tasks in C++. With our knowledge of the C++ built-in function `[sizeof()]` this function helps us to calculate the memory size of datatypes. As programmers we know that if it is an array that means it contains elements of the same datatype. So, which means if the array contains 10 elements of integers then the size of the array will be 10*4 bytes since one integer has a memory space of 4 byte. Now we can calculate the size of the array using the `[sizeof()]` function. This can be done as follows.

```
#include <iostream>
#include <vector>
using namespace std;
//#include "sqlite/sqlite3.h"
int main(){
    int arr[6] = { 10, 20, 30, 40 };
    cout<<"The size of array is: "<< sizeof(arr)/ sizeof(arr[0])<<endl;
    // OR
cout<<"The size of array is: "<< sizeof(arr)/ sizeof(int)<<endl;
// OR
cout<<"The size of array is: "<< sizeof(arr)/ sizeof(100)<<endl;
return 0;
}
```

NB: `cout<<"The size of array is: "<< sizeof(arr)/ sizeof(arr[0])<<endl;` is the safest method of getting the size of an array.

Output:

```
The size of array is: 6
The size of array is: 6
The size of array is: 6

Process finished with exit code 0
```

Indexing an array

We want to create two function. The first function will initialise an array using random numbers that are less than 20 and the second function will print all the elements that are in the array.

First function (inserting the values in the array using indexes)

```
#include <iostream>
#include <vector>
using namespace std;
//#include "sqlite/sqlite3.h"
void pushing(int arr[10]){
    srand(23);
    for(int i =0;;){
        if(i==9){break;}
        arr[i] = rand()%20;
        i++;
    }
}
```

Second function (printing the values of an array)

```
void array_print(int arr[10]){
    //int n = sizeof(arr)/ sizeof(arr[0]);
    cout<<"The array of random numbers is as follows: "<<endl;
    cout<<"[ ";
    for(int i=0;;){
        if(i==8){
            cout<<arr[i]<<" ] ";
            break;
        }
        cout<<arr[i]<<", ";
        i++;
    }
    cout<<endl;
}
```

Main function

```
int main(){
    int arr[10];
    pushing(arr);
    array_print(arr);
    return 0;
}
```

When all those functions are called in the main as above the output will be

output:

```
The array of random numbers is as follows:
[ 13, 16, 10, 5, 14, 8, 15, 14, 9 ]

Process finished with exit code 0
```

Filtering Output

Suppose we want to print all prime numbers in the array :

```
int arr[] ={13, 16, 10, 5, 14, 8, 15, 14, 9 ,7 ,2};
```

we can do it as follows:

```
#include <iostream>
#include <vector>
using namespace std;
//#include "sqlite/sqlite3.h"

int main(){
    int arr[] ={13, 16, 10, 5, 14, 8, 15, 14, 9,7,2};
    int n = sizeof(arr)/ sizeof(arr[0]);
    int i =0;
    do{
        int countf =0;
        for(int j=1;j<=arr[i];j+=1){
            if(arr[i]%j==0){
                countf +=1;
            }
        }
        if(countf==2) {
            cout << arr[i] << ": is a prime number" << endl;
        }
        i++;
    }while (i<n);
    return 0;
}
```

Output:

```
The array of random numbers is as follows:
[ 13, 16, 10, 5, 14, 8, 15, 14, 9 ]

Process finished with exit code 0
```

Printing elements of an array in reverse

Suppose we want to print the array `int arr[] ={13, 16, 10, 5, 14, 8, 15, 14, 9,7,2};` in reverse. We can do it as follows:

```
#include <iostream>
#include <vector>
using namespace std;
//#include "sqlite/sqlite3.h"

int main(){
    int arr[] ={13, 16, 10, 5, 14, 8, 15, 14, 9,7,2};
    int n = sizeof(arr)/ sizeof(arr[0]);
    int i =n-1;
    cout<<"The reversed array is as follows: "<<endl;
    cout<<"{ ";
    do{
        if(i==0){
            cout<<arr[i]<<" }";
            break;
        }
       cout<<arr[i]<<", ";
        i--;
    }while (true);
    cout<<endl;
    return 0;
}
```

Output:

```
The reversed array is as follows:
{ 2, 7, 9, 14, 15, 8, 14, 5, 10, 16, 13 }

Process finished with exit code 0
```

Searching for an element in an array

Task: we want to write a simple programme that will loop through elements of the following array, the user must enter the age that they are looking for in an array of ages. The array is as follows.

```
const int Ages[] ={21, 23, 25, 67, 54, 38, 23 , 20, 19, 18, 15,22,23,25};
```

if the age is found we must prompt: “The age was found” else the prompt must prompt “The age was not found”

```
#include <iostream>
#include <algorithm>
#include <cmath>
#include <cstdlib>
#include <ctime>
#include <afxres.h>
#include <fstream>
```

```
using namespace std;
//#include "sqlite/sqlite3.h"
int Ages[] ={21, 23, 25, 67, 54, 38, 23 , 20, 19, 18, 15,22,23,25};
bool searchAge(int arr[14],int age){
    for(int i =0;i<14; i++){

        if(arr[i]!=age){
            continue;
        }else{
            return true;
        }
    }

    return false;
}
int main(){
    cout<<"Enter the age that you are looking for in the array"<<endl;
    int age;
    cin>>age;
    switch(searchAge(Ages,age)){
        case 1:
            cout<<"The age of: "<<age<<" was found in the array of ages!"<<endl;
            break;
        case 0:
            cout<<"The age of: "<<age<<" was not found in the array of
ages!"<<endl;
            break;
        default:
            break;
    }
    return 0;
}
```

Finding the index of an element in an array

Task: suppose we want to print the index of the age that we are searching for in an array of Ages if the age is found, if not we want to pop up a message that says there's no element that you are looking for in this array. This can be done as follows.

```
#include <iostream>
#include <algorithm>
#include <cmath>
#include <cstdlib>
#include <ctime>
#include <afxres.h>
#include <fstream>
using namespace std;
//#include "sqlite/sqlite3.h"
int Ages[14] ={21, 23, 25, 67, 54, 38, 23 , 20, 19, 18, 15,22,23,25};
int main(){
    int index = -100;
    cout<<"Enter the age that you are looking for in the array"<<endl;
    int age;
    cin>>age;
    for(int i=0;i<14;i++){
        if(age == Ages[i]){
            index =i;
            break;
```

```
        }else{
            index -=1;
        }
    }
    if(index>=0){
        cout<<"The age of: "<<age<<" was found on index: "<<index<<" the array of
ages!"<<endl;
    }else{
        cout<<"The age of: "<<age<<" was not found in the array of ages!"<<endl;
    }
    return 0;
}
```

C++ program that calculate the average age of the following array of Ages.

```
int Ages[14] ={21, 23, 25, 67, 54, 38, 23 , 20, 19, 18, 15,22,23,25};
```

The program is as follows.

```
#include <iostream>
#include <algorithm>
#include <cmath>
#include <cstdlib>
#include <ctime>
#include <afxres.h>
#include <fstream>
using namespace std;
//#include "sqlite/sqlite3.h"
int Ages[] ={21, 23, 25, 67, 54, 38, 23 , 20, 19, 18, 15,22,23,25};
int main(){
    int average_Age;
    int sum_Age = 0;
    int i=0;
    while(i<sizeof(Ages)/sizeof(int)){
        sum_Age+=Ages[i];
        i++;
    }
    cout<<sum_Age;
    average_Age=sum_Age/(sizeof(Ages)/sizeof(average_Age));
    cout<<"The average age in an array of Ages is: "<<average_Age<<endl;
    return 0;
}
```

Or can be done simply as follows as well

```
#include <iostream>
#include <algorithm>
#include <cmath>
#include <cstdlib>
#include <ctime>
#include <afxres.h>
#include <fstream>
using namespace std;
//#include "sqlite/sqlite3.h"
int Ages[14] ={21, 23, 25, 67, 54, 38, 23 , 20, 19, 18, 15,22,23,25};
int main(){
    int n=0;
    int sum_Age = 0;
```

```
    int i;
    for(i=sum_Age;i<14;i++){
        n++;
        sum_Age+=Ages[i];
    }
    cout<<"The average age in an array of Ages is: "<<sum_Age/n<<endl;
    return 0;
}
```

Multi-dimensional Arrays

In C++, you can create an array known as multi-dimensional array. On the previous examples we were looking at a one-dimensional array, multi means more than one. Which means if an array is a multi-dimension, we are expecting it to have rows and columns, We have 2-dimensional array, three-dimensional array and so on. We are going to learn on how to declare initialise and accessing elements of arrays with different dimensions.

2-dimensional array

A two-dimensional array can be declared as follows.

```
datatype array_name [rows][columns]
```

a two-dimensional array can declare and be initialised at the same time as follows.

```
datatype array_name[rows] [columns] = {values}
```

Example on how to declare, then initialise values after in an array of integers:

```
int main(){
    int my_arr[3][2];
    my_arr[0][0] =9;
    my_arr[0][1] =19;
    my_arr[1][0] =29;
    my_arr[1][1] =39;
    my_arr[2][0] =49;
    my_arr[2][1] =59;
    return 0;
}
```

So, having the above arrays we can print them in a matrix form by accessing each element by its respective row and column. This can be done as follows:

```
#include <iostream>
#include <algorithm>
#include <cmath>
#include <cstdlib>
#include <ctime>
#include <afxres.h>
#include <fstream>
```

```cpp
using namespace std;
//#include "sqlite/sqlite3.h"

int main(){
    int my_arr[3][2];
    my_arr[0][0] =9;
    my_arr[0][1] =19;
    my_arr[1][0] =29;
    my_arr[1][1] =39;
    my_arr[2][0] =49;
    my_arr[2][1] =59;
    cout<<"my_array ={\n";
    for(int i=0;i<3;i++){
            cout<<"[";
        for(int j=0;j<2;j++){
            cout<<my_arr[i][j]<<" ";
        }
        cout<<"]"<<endl;
    }
    cout<<"};";
}
```

Output:

```
my_array ={
[9 19 ]
[29 39 ]
[49 59 ]
};
Process returned 0 (0x0)   execution time : 0.076 s
Press any key to continue.
```

Declaration, initialisation and printing the elements of a 2D array

You can declare and initialise a two-dimensional array in many ways.

First method:

```cpp
#include <iostream>
#include <algorithm>
#include <cmath>
#include <cstdlib>
#include <ctime>
#include <afxres.h>
#include <fstream>
```

```cpp
using namespace std;
//#include "sqlite/sqlite3.h"
int main(){
    int Ages[3][4] ={21, 23, 25, 67, 54, 38, 23 , 20, 19, 18, 15,22};
    // this is an array of 3 rows and 4 columns
    int i,j;
    for(j=0;j<3;j++){
        for(i=0;i<4;i++){
            cout<<Ages[j][i]<<"\t";
        }
        cout<<endl;
    }
    return 0;
}
```

In the declaration of array of Ages, we just define the dimension of 4 rows 2 columns. So, the first 4 values will be of the first column. And the second 4 values will be of the next column. The output will be as follows:

```
21      23      25      67
54      38      23      20
19      18      15      22

Process returned 0 (0x0)   execution time : 0.205 s
Press any key to continue.
```

Second Method:

Better way to initialise this array with same array elements as above.

```cpp
#include <iostream>
#include <algorithm>
#include <cmath>
#include <cstdlib>
#include <ctime>
#include <afxres.h>
#include <fstream>
using namespace std;
//#include "sqlite/sqlite3.h"
int main(){
    int Ages[3][4] ={
        {21, 23, 25, 67},
        {54, 38, 23 , 20},
         {19, 18, 15,22}
        };
```

```
    // this is an array of 3 rows and 4 columns
    int i,j;
    for(j=0;j<3;j++){
        for(i=0;i<4;i++){
            cout<<Ages[j][i]<<"\t";
        }
        cout<<endl;
    }
    return 0;
}
```

Output:'

```
21      23      25      67
54      38      23      20
19      18      15      22

Process returned 0 (0x0)   execution time : 0.205 s
Press any key to continue.
```

Printing the elements of a 2D array using the sizeof() function to determine the (maximum index - 1)

```
#include <iostream>
#include <algorithm>
#include <cmath>
#include <cstdlib>
#include <ctime>
#include <afxres.h>
#include <fstream>
using namespace std;
//#include "sqlite/sqlite3.h"
int main(){
    int Ages[3][4] ={
        {21, 23, 25, 67},
        {54, 38, 23 , 20},
         {19, 18, 15,22}
        };
    // this is an array of 3 rows and 4 columns
    int i,j;
    for(j=0;j<sizeof(Ages)/sizeof(Ages[0]);j++){
        for(i=0;i<sizeof(Ages[0])/sizeof(Ages[0][0]);i++){
            cout<<Ages[j][i]<<"\t";
        }
```

```
        cout<<endl;
    }
    return 0;
}
```

Output:

```
21      23      25      67
54      38      23      20
19      18      15      22

Process returned 0 (0x0)   execution time : 0.205 s
Press any key to continue.
```

Task: we want to write a program that will add integer values in the 2D array by randomising them using the rand() function. The array must have 4 rows and 5 columns, and these values must be printed smartly on the screen.

```
#include <iostream>
#include <algorithm>
#include <cmath>
#include <cstdlib>
#include <ctime>
#include <afxres.h>
#include <fstream>
using namespace std;
//#include "sqlite/sqlite3.h"
int main(){
    int myarray[4][5];
    srand(23);
    for(int i=0;i<sizeof(myarray)/sizeof(myarray[0]);i++){
        for(int j=0;j<sizeof(myarray[0])/sizeof(myarray[0][0]);j++){
        myarray[i][j]=round((rand()%30));
    }
    }
    for(int i=0;i<sizeof(myarray)/sizeof(myarray[0]);i++){
        for(int j=0;j<sizeof(myarray[0])/sizeof(myarray[0][0]);j++){
        cout<<myarray[i][j]<<"\t";
    }
    cout<<endl;
    }

    return 0;
}
```

Output:

```
23      26      10      15      4
28      15      14      9       28
21      14      6       28      9
1       16      21      5       19

Process returned 0 (0x0)   execution time : 0.076 s
Press any key to continue.
```

Searching for an element in a 2D array:

We want to write a program that will search for an integer that is in a 2D array if the integer is found we want to see the indexes of the integer position of the that integer in terms of row number and position number. We are going to modify the above program that focuses on random number. And the programme must print the array in a matrix for so that we can be able to verify if the indexes are correct.

```
#include <iostream>
#include <algorithm>
#include <cmath>
#include <cstdlib>
#include <ctime>
#include <afxres.h>
#include <fstream>
using namespace std;
//#include "sqlite/sqlite3.h"
int main(){
    int myarray[4][5];
    srand(23);
    for(int i=0;i<sizeof(myarray)/sizeof(myarray[0]);i++){
        for(int j=0;j<sizeof(myarray[0])/sizeof(myarray[0][0]);j++){
        myarray[i][j]=round((rand()%30));
        }
    }
    for(int i=0;i<sizeof(myarray)/sizeof(myarray[0]);i++){
        for(int j=0;j<sizeof(myarray[0])/sizeof(myarray[0][0]);j++){
        cout<<myarray[i][j]<<"\t";
     }
      cout<<endl;
    }
    int row =-1, choice;
    cout<<"Enter the number that you are looking for in this array"<<endl;
    cin>>choice;
    int column =row;
    bool found =false;
        for(int i=0;i<sizeof(myarray)/sizeof(myarray[0]);i++){
```

```
        for(int j=0;j<sizeof(myarray[0])/sizeof(myarray[0][0]);j++){
               if(choice==myarray[i][j]){
                    found = true;
                    row = i;
                    column =j;
                    goto label;
                }
             }
        }
    label: if(found){
        cout<<"The number :"<<choice<< " was found on position [";
        cout<<row<<"]["<<column<<"]"<<endl;
    }else{
         cout<<"The number :"<<choice<< " was not found !!";
    }
    return 0;
}
```

Output if found:

```
23      26      10      15      4
28      15      14      9       28
21      14      6       28      9
1       16      21      5       19
Enter the number that you are looking for in this array
28
The number :28 was found on position [1][0]

Process returned 0 (0x0)   execution time : 14.739 s
Press any key to continue.
```

Output if not found:

```
"C:\Users\garid\Desktop\C++ Profesional\bin\Debug\C++ Profesional.exe"
23      26      10      15      4
28      15      14      9       28
21      14      6       28      9
1       16      21      5       19
Enter the number that you are looking for in this array
33
The number :33 was not found !!
Process returned 0 (0x0)   execution time : 3.163 s
Press any key to continue.
```

3 – dimensional Array

We are not going to talk much on this one. We are only going to show how do we declare and initialise a 3D array and how are we going to loop through elements of a 3D array, otherwise the idea is the same with the 2D array.

Declaration, Initialisation and printing elements of a 3D array.

Suppose we have an array of integers that is called test, and test can be declared and being initialised as follows.

```
//
int test[2][3][4] = {3, 4, 2, 3, 0, 3, 9, 11, 23, 12, 23, 2, 13, 4, 56, 3, 5,
9, 3, 5, 5, 1, 4, 9};
//
```

Test can also be declared and being initialised as follows.

```
int test[2][3][4] = {
                    { {3, 4, 2, 3}, {0, -3, 9, 11}, {23, 12, 23, 2} },
                    { {13, 4, 56, 3}, {5, 9, 3, 5}, {3, 1, 4, 9} }
                };
```

Explanation

```
int test[2][3][4]
```

test has 2 major rows which are

```
 //row 1 { {3, 4, 2, 3}, {0, -3, 9, 11}, {23, 12, 23, 2} },
 // row 2 { {13, 4, 56, 3}, {5, 9, 3, 5}, {3, 1, 4, 9} }
```

This makes another dimension of test. Inside row 1 there is a two-dimensional array the same applies to row 2. These inner rows have the following dimensions which is 3 rows and 4 columns each. So now we can print out all the values that are in each inner row separately as follows:

```
#include <iostream>
#include <algorithm>
#include <cmath>
#include <cstdlib>
#include <ctime>
#include <afxres.h>
#include <fstream>
using namespace std;
//#include "sqlite/sqlite3.h"
int main(){
        int test[2][3][4] = {
                    { {3, 4, 2, 3},
                    {0, -3, 9, 11},
                    {23, 12, 23, 2} }, // second row
                    { {13, 4, 56, 3},
                     {5, 9, 3, 5},
```

```
                    {3, 1, 4, 9} }
                };
                cout<<"The first part"<<endl;
      for(int i =0; i<sizeof(test)/sizeof(test[0]);i++){
        // the outer loop for the third dimension
        cout<<"[ \n";
        for(int j=0;j<sizeof(test[0])/sizeof(test[0][0]);j++){
            //the first inner loop we are now in the 2d array of test

            for(int k=0;k<sizeof(test[0][0])/sizeof(test[0][0][0]);k++){
            //the seconde inner loop we are now in the 1d array of test
             cout<<test[i][j][k]<<"\t";
          }
         cout<<endl;
        }
        if(i==sizeof(test)/sizeof(test[0])-1){
               cout<<"]\n";
             break;
        }else{
            cout<<"]\n\n The second part"<<endl;
        }
      }

    return 0;
}
```

Output:

```
The first part
[
3       4       2       3
0       -3      9       11
23      12      23      2
]

 The second part
[
13      4       56      3
5       9       3       5
3       1       4       9
]

Process returned 0 (0x0)   execution time : 0.120 s
Press any key to continue.
```

Strings as array of characters.

String is a collection of characters. In other words, a string is an array of character.

We want to create a program that will read a string from the user when we declare a variable that hold our string with char datatype this can done as follows.

```
#include <iostream>
#include <algorithm>
#include <cmath>
#include <cstdlib>
#include <ctime>
#include <afxres.h>
#include <fstream>
using namespace std;
//#include "sqlite/sqlite3.h"
int main(){
	char name[20];
	cout<<"Enter your name"<<endl;
	cin>>name;
	cout<<"Your name is: "<<name<<" it contains: "<<string(name).size()<<"
character(s)\n";
    return 0;
}
```

So now we can loop through all the characters of the string name if we want.

Array and functions.

We have covered this on the beginning of this section, now we want to do two more examples one for printing a character of a string and the other for finding the largest number in a 2D array.

Printing characters of a string

Consider an example of getting a string from the user after declaring the string with a char datatype. Now we want to modify so that we can print all the characters on the screen. We can do it as follows.

```
#include <iostream>
#include <algorithm>
#include <cmath>
#include <cstdlib>
#include <ctime>
#include <afxres.h>
#include <fstream>
using namespace std;
//#include "sqlite/sqlite3.h"
void stringPrint(char name[]);
int main(){
	char name[20];
	cout<<"Enter your name"<<endl;
	cin>>name;
	 stringPrint(name);

    return 0;
}
void stringPrint(char name[]){
```

```
    for(int i=0;;){
        if(i==string(name).length()){
            goto label;
        }
        cout<< name[i] <<",\t";
        i++;
      }
      label:
      cout<<endl;
}
```

Output:

```
Enter your name
crispen
c,      r,      i,      s,      p,      e,      n,

Process returned 0 (0x0)   execution time : 4.508 s
Press any key to continue.
```

A function to array of finding the sum of ages in a 1D array

We have the following array of age we want to calculate the sum of all ages in the array Ages this can be done as follows.

```
   int Ages[12]={21, 23, 25, 67, 54, 38, 23 , 20, 19, 18, 15,22};
```

```
#include <iostream>
#include <algorithm>
#include <cmath>
#include <cstdlib>
#include <ctime>
#include <afxres.h>
#include <fstream>
using namespace std;
//#include "sqlite/sqlite3.h"
int sum_Ofage(int age[]);
int main(){
        int Ages[12]={21, 23, 25, 67, 54, 38, 23 , 20, 19, 18, 15,22};
        cout<<"The sum of the ages in the array Ages is: "<< sum_Ofage(Ages)<<
endl;

    return 0;
}
int sums =0;
int sum_Ofage(int age[]){
    for(int i=0;i<12;i++){
        sums+=age[i];
```

```
    }
    return sums;
}
```

Output:

```
The sum of the ages in the array Ages is: 345

Process returned 0 (0x0)   execution time : 0.716 s
Press any key to continue.
```

Finding the largest age in an array in a 2D array.

We have the following array of age we want to find the largest age in this array using functions and arrays

```
  int Ages[3][4] ={21, 23, 25, 67, 54, 38, 23 , 20, 19, 18, 15,22};
```

We can do It as follows:

```
#include <iostream>
#include <algorithm>
#include <cmath>
#include <cstdlib>
#include <ctime>
#include <afxres.h>
#include <fstream>
using namespace std;
//#include "sqlite/sqlite3.h"
void large_Ofage(int age[3][4]);
int main(){
        int Ages[3][4] ={21, 23, 25, 67, 54, 38, 23 , 20, 19, 18, 15,22};
       large_Ofage(Ages);
    return 0;
}
int largest;
void large_Ofage(int age[3][4]){
     largest = age[0][0];
     int row=0,column =0;
    for(int i=0;i<3;i++){
         for(int j=0;j<4;j++){
       if(age[i][j]>=largest){
        largest = age[i][j];
        row =i;column =j;
       }else{
           continue;
```

```
        }
    }
    }
    cout<<"The largest number in the array is: "<<largest;
    cout<<" which is on index ["<<row<<"]["<<column<<"]"<<endl;

}
```

---------- *END* ----------