Arrays in C++

Arrays in C++ is a collection of elements of similar data type like int, char, float, double, etc, that are stored using the index value and can easily be accessed by index value only. It implements to store all the instances of variables into one single variable. In C++, an array can be declared using three various methods – by specifying the size of an array, by initializing array elements directly and by specifying arrays size with its elements.

To let the data processed using any application, we first need to bring the data into the application. This means there should be some space in the application where the value should be stored until the program runs. In order to serve the purpose of storing the values, the programming language offers us variable. Variables are used to store the values so that they could be used by the application to generate the expected outcome. As variables store, some value, occupy space in the memory allocated to the application. So the optimal approach of coding is to ensure the usage of the variable should be as low as possible. In order to mitigate the memory allocation issue due to the creation of a high number of variables, the concept of array came into existence. The array can be considered as the list of values that belongs to the same datatype. In this article, we are going to learn about the array using the C++ programming language.

How Do Arrays work in C++?
Below is the explanation of how arrays work:

- **The role of the array is to store the values of the same datatype. It is supposed to work the same way as that of the variable and the only additional point it has over variable is, it is capable of holding several values at the same time. When we create the array in C++ or any of the programming language, we have to state the number of variables that we want to store in the array.**
- **It has to be noted that the size of the array remains fixed throughout the application runtime and cannot be changed dynamically. Once the size of the array is defined, we can store the same count of values in it. If the data type of the array is defined as an integer, it will not accept any value that is not an integer. In order to locate the value held by the array one will need to use the index.**
- **For instance, if the array is capable of holding two values then the second value will be stored at the one index of array as the index of array begins with zero. In the next section, we will learn array creation.**

How to Create Arrays in C++?
Below explanation says that how to create arrays in c++:

The approach of creating the array is exactly similar to variable creation. The first step is to declare the array. Once the array is declared, we can either initialize the array at the same time or it could be initialized later. While declaring the array we have to mention three things: datatype of the array, the name of the array and its size. Below is the syntax that shows how to merely declare the array.

Syntax:

```
Datatype array_name[size];
```

```
Ex. int  first_array[10];
```

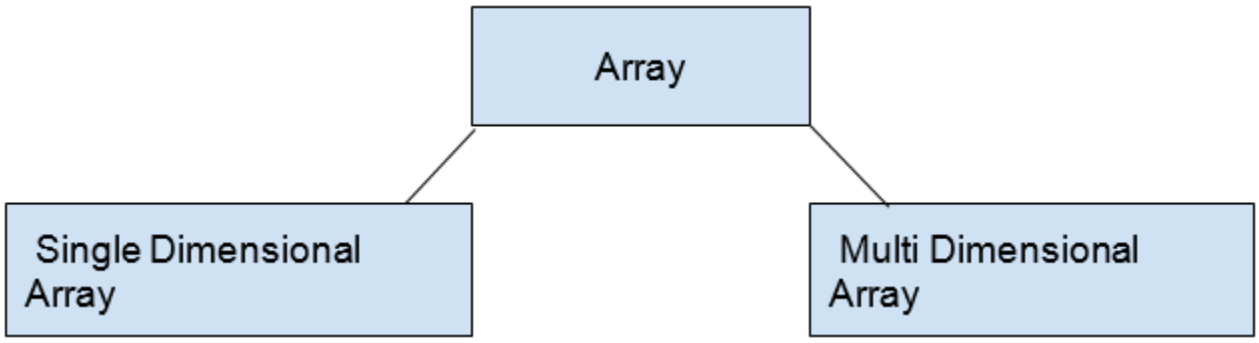
The array defined here can ten integer values. The name of the array is first_array and the number defined inside the large bracket states the size of the array. Now let's see how to declare and initialize the variable at the same time.

```
Int first_array[4] = { 1,2,3,4}
```

```
Int first_array[]= {1,2,3,4,5,6}
```

In the above example, we have seen that the array that has defined the size 4 has accepted the 4 values. If one will try to submit more than 4 values, it will throw an error. Also, if you do not specify the size of the variable you can store as much value as you want.

Types of Arrays in C++



In C++ programming language we do have mainly two types of variables: Single Dimensional Arrays and multidimensional Arrays. The single-dimensional stores the values hold the values in the form of the list while the multidimensional array store the value in the matrix. Below we will see each of the types using an example.

1. Single Dimensional Array

The single-dimensional array may be defined as the type of array that is capable to hold the values of the same datatype in the form of a list. It is the simplest form of array as it doesn't require much effort to define and initialize such array. It can be defined as int a[10], where int is the data type is the array name and the size of the array is 10. The example below will make things more clear.:

```
#include <iostream>

using namespace std;

void main()

{

int val_array[3];

int m=1,c;

cout<<"Please enter three numbers that you want to multiply"<<endl;

for(c=0;c<3;c++) {

cin>>val_array[c];

m = m*val_array[c];  }
```

```
cout<<"The multiplication of these three numbers is = "<<m;
}
```

The above program is written to accept three values from the user and then those values will be processed to generate the multiplication value of all of them. The array user here is name val_array and the array is capable to hold three values. The loop is used to intake the value in the array and then they were multiplied. The end result of the multiplication has been stored in the int_val variable. As the function is void in nature, it will not return any value.

2. Multidimensional array

The multidimensional array may be defined as the array that holds the values in the way a matrix does. The two-dimensional array is used very often and with the increase in the size of dimension, the array gets complicated. For instance, it is easy to work with a two-dimensional array rather than working with a three-dimensional array. The two-dimensional array needs two sizes to be defined for one dimension each. The two-dimensional array can be in the program as int a[5][3]. This array will be able to hold the value in the form of a matrix that has 5 rows and three columns. Let us understand this with the help of an example.

```
#include <iostream>
using namespace std;
int main()
{
```

```
int val_array[5][5];
int count_rows,count_cols,counter1,counter2;
cout<<"Please enter the size of the rows and columns that you wnant to input: ";
cin>>count_rows>>count_cols;
cout<<"PLease enter the values for matrics in row wise manner"<<endl;
for(counter1=0;counter1<count_rows;counter1++)
for(counter2=0;counter2<count_cols;counter2++)
cin>>val_array[counter1][counter2];
cout<<"The matrix will be as follows"<<endl;
for(counter1=0;counter1<count_rows;counter1++)
{
for(counter2=0;counter2<count_cols;counter2++)
cout<<val_array[counter1][counter2]<<"  ";
cout<<endl;
}
return 0;
}
```

In this program, we have used a two-dimensional array. The way array was defined using two sizes, it states that the array used is two dimensional. If there would have

been three sizes then the array would be three dimensional. The user is asked to input the number of rows and columns they want in the matrix. Once the numbers are specified by the user, they are asked to input the values that they want in the rows and columns of the matrix. Here the user has submitted 2 3 which means they want two rows and three columns in the matrix. Know they had to submit six values as the matrix with two rows and three columns have six values. When all the values have been submitted, they have been represented in the form of a matrix. The entire program is implemented using the two-dimensional array.

The General Syntax of Array Declaration is –

```
<Type of array> <array name> <[Size of array]>
```

Example –

```
int pincode_of_cities[50]
```

The size of the array must be an integer constant value which must be greater than 0. The type of array can be any data types valid in C++.

Initializing of C++ Array
There are two basic methods of initializing an array –

Method 1: At declaration time

```
int pincode[5] = {123, 000, 342, 678, 654};
```

OR

```
int pincode[] = {123, 000, 342, 678, 654};
```

Method 2: Using a loop

```
int number[5];

for(int i = 0; i<sizeof(pincode); i++)

number[i] = i;
```

This is a one-dimensional array or a 1D array. The second type of array is a multi-dimensional array which we will discuss little later.

First of all, let us look at how we can access values from an array. Few of the following methods are given below –

Accessing Values of an Array

Name of the array[index]: This will return the value at the index position mentioned.

```
#include <iostream>

using namespace std;

int arr[5] = {10,20,30,40,50};
```

```
int main ()
{
cout << arr[3];
return 0;
}
```

In order to print all the values in an array –

```
#include <iostream>
using namespace std;
int arr[5] = {10,20,30,40,50};
int i;
int main ()
{
for ( i=0 ; i<5 ; i++ )  {
cout << arr[i];  }
return 0;  }
```

Example:

```
#include <iostream>
using namespace std;
```

```
int main()

{

int num[5], sum = 0;

cout << "Enter 5 natural numbers: ";

//  Store numbers

//  Then find sum

for (int i = 0; i < 5; ++i)     {

cin >> num[i];

sum += num[i];   }

cout << "Sum = " << sum << endl;

return 0;

}
```

Multidimensional Array

The above is the declaration for a single dimensional array. The second type of array is the multidimensional array and is also known as rectangular arrays in C++. Depending on the requirement, it can be a two-dimensional array or a three-dimensional array. The values are stored in a table format, also known as a matrix in the form of rows and columns.

The syntax to declare a multidimensional array is –

<data type> <name of array>[number of rows][number of columns] int

two_dim[2][2];

This means that the above array has –

- **2 rows**
- **2 columns**

The above array can be initialized in the following way –

Method 1:

```
#include <iostream>

using namespace std;

int main()

{

int arr[3][3] =

{

{1, 5, 15},

{44, 0, 23},

{29, 41, 85}  };  //declaring and initializing at the same time

//traversing through the array

for(int i = 0; i < 3; ++i)

{
```

```
for(int j = 0; j < 3; ++j)

{

cout<< arr[i][j]<<" ";

}

cout<<"\n"; //this will take to a new line once the all the columns of

the                                    //particular row has been traversed

}

return 0;

}
```

Output:

```
1       5       15

44      0       23

29      41      85
```

Method 2:

```
#include <iostream>

using namespace std;

int main()

{
```

```
int arr[3][3];  //declaring a 2D array

arr[0][0]=5;  //initializing the array

arr[0][1]=10;

arr[0][2]=15;

arr[1][0]=20;

arr[1][1]=30;

arr[1][2]=40;

arr[2][0]=50;

arr[2][1]=60;

arr[2][2]=70;

//traversing through the elements in the array

for(int i = 0; i < 3; ++i)   {

for(int j = 0; j < 3; ++j)

{cout<< arr[i][j]<<" " ;    }

cout<<"\n"; //this will take to a new line once the all the columns of

the                                    //particular row has been traversed

}

return 0;

}
```

Example:

```
#include <iostream>

using namespace std;

const int city = 2;

const int week = 2;

int main() {

int temp[city][week];

cout << "Enter temp for city \n";

// Insert values

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

for(int j = 0; j < week; ++j)   {

cout << "city " << i + 1 << ", Week Day " << j + 1 << " : ";

cin >> temp[i][j];  }}

// Access values

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

for(int j = 0; j < week; ++j)   {

cout << "city " << i + 1 << ", Week Day " << j + 1 << " = " << temp[i][j] << endl;

}return 0; }
```

What is a Struct in C++?

A STRUCT is a C++ data structure that can be used to store together elements of different data types. In C++, a structure is a user-defined data type. The structure creates a data type for grouping items of different data types under a single data type.

```
struct Book {
        string title؛
        string author؛
        double price؛
        int numberOfPages؛
}book1;


}book1,book2,book3;
```

```
  book1.title = "C++ for beginners";
   book1.author = "Mhamad ";
   book1.price = 9.99؛
   book1.numberOfPages = 420؛


cout <<" Book 1 ---------- \n؛"
   cout <<" Title :" << book1.title <<endl;
   cout <<" Author :"<<  book1.author <<endl;
   cout <<" Price :"<< book1.price <<endl ;
   cout <<" Number of pages :" <<book1.numberOfPages;
```

```
#include <iostream<

    using namespace std؛
    struct Book}
    string title؛
    string author؛
    double price؛
    int numberOfPages؛

    {book1؛
    int main()
    }
    book1.title = "C++ for beginners؛"
    book1.author = "Mhamad Harmush؛"
    book1.price = 9.99؛
    book1.numberOfPages = 420؛
    cout << "Title: " << book1.title << "\n؛"
    cout << "Author: " << book1.author << "\n؛"
    cout << "Price: " << book1.price << "$\n؛"
    cout << "Number of pages: " << book1.numberOfPages << "\n؛"
    return 0؛
    {
```

Example: Write a program in c++language to read the student's name, stage, and age for ten students, and then print them by using structure

```
# include <iostream>
        using namespace std;
        struct stud {
        string name;
        int  age;
        int stage;

        }student1[10] ;
        int main()
        {
                int i;
        for(i=0;i<10;i++){
                cout<<"student "<<i<<endl;
           cout<<"name:";
        cin>>student1[i].name;
                cout<<endl<<"age:";
       cin>>student1[i].age;
                cout<<endl<<"stage:";
         cin>>student1[i].stage ;
}
        for(i=0;i<10;i++){
```

```
cout<<endl<<"**************student"<<"\t"<<i<<endl;
        cout << "name: " << student1[i].name << "\n";
        cout << "age: " << student1[i].age << "\n";
        cout << "stage: " << student1[i].stage << "\n";
}
        return 0;
        }
```

H.W:1. Write a program in c++language to read the name and three grades for five students and print the name and average

2. Write a program in c++language to read the names of 5 students and their ages, and print the name and age if it is greater than 18 by using a) structure b) structure and function

C++ Functions

A function is a block of code which only runs when it is called.

You can pass data, known as parameters, into a function.

Functions are used to perform certain actions, and they are important for reusing code: Define the code once, and use it many times.

Create a Function

C++ provides some pre-defined functions, such as main(), which is used to execute code. But you can also create your own functions to perform certain actions.

To create (often referred to as *declare*) a function, specify the name of the function, followed by parentheses **()**:

Syntax

```
void  myFunction() {
  // code to be executed
}
```

Example Explained

- myFunction() is the name of the function
- void means that the function does not have a return value. You will learn more about return values later in the next chapter
- inside the function (the body), add code that defines what the function should do

Call a Function

Declared functions are not executed immediately. They are "saved for later use", and will be executed later, when they are called.

To call a function, write the function's name followed by two parentheses () and a semicolon ;

Ex :write program by using function to find sum of two numbers

```
# include<iostream>

Using namespace std;
```

```
Void sum ()

{

Int x,y

Cout<<"enter the two numbers";

Cin>>x>>y;

Cout<<"sum="<<x+y;

}
```

```
Int main ()
{
Sum( );
Return 0;
}
```

Example: write program by using function to find the value of $z=x^2+y^2$

```
# include<iostream> # include<cmath>
Using namespace std;
Void find ()                    الداله
{   Int x,y,z;
Cout<<"enter the two numbers";
Cin>>x>>y;
Z=pow(x,2)+pow(y,2);
Cout<<"z="<<z;   }
Int main ()
{
find( );          استدعاء الداله
Return 0;
}
```

Ex :write program using function to find sum of two numbers

```
# include<iostream>
Using namespace std;
int sum ( )
```

```
{
Int x,y
Cout<<"enter the two numbers";
Cin>>x>>y;
Rturn(x+y);
}
انتهت الداله


Int main ()
{
Cout<< Sum( );
Return 0;
}
```

Ex :write program using function to find the value s=1+3+5+7….n and return the value

```
# include<iostream>
Using namespace std;
int sum ( )
{
Int s=0,n;
Cout<<"enter the number";
Cin>>n;
For (int i=1;i<=n;i+=2)
   s=s+i;
```

```
Rturn(s);                    ارجاع النتيجه return نستخدم هنا
}
انتهت الداله
Int main ()
{
Cout<< Sum( );
Return 0;
}
```

امثلة اضافية:

Ex: Write program in c++ by using function to count the iterations of number five

```
# include<iostream >
Int x[10],z=0,i;
void check(int x[])
{
For(i=0;i<4;i++)
If (x[i]==5)
Z++;
Cout<<z;
}
Int main (){
Cout<<"enter the elements of array ";
For(i=0;i<4;i++)
Cin>>x[i];
check(x);
Return 0;
}
```

Ex: Write program in c++ by using function to print the elements above the main diagonal

```
# include<iostream >
Using namespace std;
Int x[10][10] ,I,j;
void check(int x[][10])
{
For(i=0;i<3;i++){
For(j=0;j<3;j++){
If (i<j)
Cout<<x[i][j]<<"  ";}
}
}
Int main (){
Cout<<"enter the elements of array ";
For(i=0;i<3;i++){
For(j=0;j<3;j++)
Cin>>x[i][j];}
check(x);
Return 0;
}
```

ex: :write program by using function to find the area of rectangle مساحه المستطيل

ex: :write program by using function to find the value s=1/1+1/2+1/3…….1/n and return the value to main program .

ex: write program by using function to check the number odd or even هل العدد زوجي ام فردي

++++++++++++++++++++++++++

C++ Functions

Ex :write program by using function with argument to find sum of two numbers

```
# include<iostream>

Using namespace std;
```

```
Void sum (int x, int y)          {

Cout<<"sum="<<x+y;

}

انتهت الداله
```

```
Int main ()

{    Int x,y

Cout<<"enter the two numbers";

Cin>>x>>y;                              or   sum(3,5)

Sum(x,y );          استدعاء الداله

Return 0;}
```

Example: write program using function with argument to find the value of $z=x^2+y^2$

```
# include<iostream> # include<cmath>

Using namespace std;

Void find (int x, int y)                    الداله
```

```
{  int z;

Z=pow(x,2)+pow(y,2);

Cout<<"z="<<z;

}

Int main ()

{

Int x,y;

Cout<<"enter the two numbers";

Cin>>x>>y;

find( x,y);          استدعاء الداله

Return 0;     }
```

Ex :write program by using function with argument to find the largest number between two numbers

Ex :write program using function with arguments to find sum of two numbers and return the result to the program

```
# include<iostream>

Using namespace std;



int sum (int x ,int y )

{

Rturn(x+y);                    ارجاع النتيجه  return  نستخدم  هنا

}

Int main ()

{
```

```
Int x,y;

Cout<<"enter the two numbers";

Cin>>x>>y;


Cout<< Sum(x,y );

Return 0;

}
```

Ex :write program using function to find the value s=2+4+6+8….n and return the value

```
# include<iostream>

Using namespace std;

int sum (int n )

{ int s=0;

For (int i=2;i<=n;i+=2)

   s=s+i;

Rturn(s);

}


Int main ()

{     Int ,n;
```

```
Cout<<"enter the number";

Cin>>n;



Cout<< Sum( n );

Return 0;

}
```

Ex: **:write program using function with arguments to find the area of circle** مساحه الدائره

ex: :write program using function with arguments to find the value s=1/1+1/3+1/5…….1/n and return the value to main program .

++

Ex: Write program in c++ by using function to find the summation of two numbers (global variables)

```
# include <iostream>

using namespace std;

int x,y,z;                              // global

void   sum()

{

Cin>>x>>y;

z=x+y;

Cout<<z;

}

int main()

{

Sum();
```

```
Cout<<x<<y<<z;
Return 0;
}
```

C++ Pointers

we can get the memory address of a variable by using the & operator:

Example

```
char X[] = "HELLO";

cout <<X;
```

Outputs the value of X

```
cout << &X;
```

Outputs the memory address

A pointer however, is a variable that stores the memory address as its value.

A pointer variable points to a data type (like int or string) of the same type, and is created with the * operator. The address of the variable you're working with is assigned to the pointer:

Example

```
int X = 12;                    // X variable of type integer
int*  ptr = &X ;          // A pointer variable, with the name ptr, that stores the address of X

cout << X << endl;        // Output the value of X (12)

cout << &X << endl;   // Output the memory address of X (0x6dfed4)

cout << ptr << endl;     // Output the memory address of X with the pointer (0x6dfed4)
```

++

ex: write program by using pointer to find the summation of two numbers

```
# include<iostream >
using namespace std;
int main()
{
        int x=3,y=5;
        int *pt=&x;
        int *pt1=&y;
        cout<<pt<<endl;
        cout<<pt1<<endl;
        cout<<*pt+*pt1;
        return 0;
}
++++++++++++++++++++++++++
```

h.w/ write program by using pointer to find th average among three degree

---

```
# include<iostream >
using namespace std;
int main()
{
        char x='F';
        char *pt=&x;
        cout<<*pt;

return 0;}
```

```
# include<iostream>
using namespace std;
int main( )
```

```
{
 int x[10],i;
int *p=x;                    //or int *p=&X[0];

for(i=0;i<5;i++)
cin>>x[i];                   /
for(i=0;i<5;i++)
cin>>*(p+i);                  /
cout<<"*****************************************"<<endl;
for(i=0;i<5;i++)
cout<<x[i];                          //
cout<<"+++++++++++++++++++++++++++++++++++++++"<<endl;

for(i=0;i<5;i++)
cout<<*(p+i);                //
return 0;
}++++++++++++++++++++++++++++


# include<iostream>
using namespace std;
int main( )
{
 int x[10],i;
int *p=x;                    //&X[0];

for(i=0;i<5;i++)
```

```
cin>>*(p+i);
//----------------*(p+i)=x[i]------------
for(i=0;i<5;i++)
if (*(p+i)%2==0)
cout<<*(p+i);
return 0;
}
```

```
# include<iostream>
using namespace std;
void find (int *p,int *p1)
{
int s;
s=*p+*p1;
cout<<s;
}
int main( )
{
int x,y;
cin>>x>>y;
find (&x,&y);
return 0;
}
```

---

h.w/ write program by using pointer and function to find th average among three degree

h.w/ write program by using pointer and function to find the largest number between two numbers

Character String

Character String is actually a one-dimensional array of characters which is terminated by a null character '\0'. Thus a null-terminated string contains the characters that comprise the string followed by a null.

The following declaration and initialization create a string consisting of the word "Hello". To hold the null character at the end of the array, the size of the character array containing the string is one more than the number of characters in the word "Hello."

Actually, you do not place the null character at the end of a string constant. The C++ compiler automatically places the '\0' at the end of the string when it initializes the array. Let us try to print above-mentioned string –

```
#include <iostream>

using namespace std;

int main () {

   char str[6] = {'H', 'e', 'l', 'l', 'o', '\0'};

   cout << "Greeting message: ";
   cout << str << endl;

   return 0;
}
```

When the above code is compiled and executed, it produces the following result –

Greeting message: Hello

```
#include <iostream>

using namespace std;

int main () {

   char str[10] ;
   cout << "enter the string: ";
  cin >>str;
```

```
  cout << "Greeting message: ";
  cout << str << endl;

  return 0;
}
```

C++ supports a wide range of functions that manipulate null-terminated strings –

Sr.No	Function & Purpose
1	**strcpy(s1, s2);** Copies string s2 into string s1.
2	**strcat(s1, s2);** Concatenates string s2 onto the end of string s1.
3	**strlen(s1);** Returns the length of string s1.
4	**strcmp(s1, s2);** Returns 0 if s1 and s2 are the same; less than 0 if s1<s2; greater than 0 if s1>s2.

```
#include <iostream>
#include <cstring>

using namespace std;

int main () {

  char str1[10] = "Hello";
  char str2[10] = "World";
  char str3[10];
  int  len ;


  strcpy( str3, str1);
```

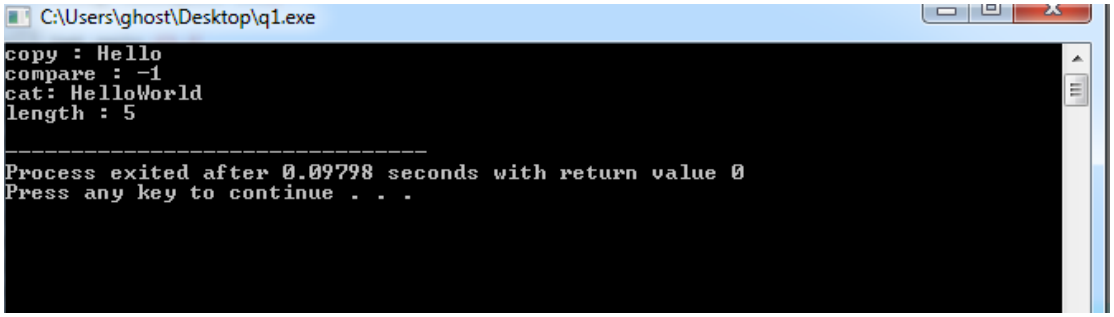
```
  cout << "copy : " << str3 << endl;

  strcat( str1, str2);
  cout << "cat: " << str1 << endl;

  len= strlen(str1);
  cout << "length : " << len << endl;

  return 0;
}
```

When the above code is compiled and executed, it produces result something as follows:

```
copy : Hello
cat : HelloWorld
length : 10
++++++++++++++++++++++++++++++++++++++++++++++++++++++++
```

```
#include <iostream>

#include <cstring>

using namespace std;

int main () {


  char str1[10] = "Hello";

  char str2[10] = "World";

  char str3[10];

  int  len,c ;


  strcpy( str3, str1);

  cout << "copy : " << str3 << endl;


  c=  strcmp( str1, str1);

  cout << "compare : " << c << endl;


   strcat( str1, str2);
```

```
  cout << "cat: " << str1 << endl;


  len= strlen(str2);
  cout << "length : " << len << endl;
  return 0;
}
```



++

Ex: Write a program to read a string and print a specific character, for example, the second

```
#include <iostream>
#include <cstring>
using namespace std;

int main()
{
 string    a;            // char  a[10];

cout<<"enter the string";
cin>>a;

cout<<”the letter:”
```

```
cout<<a[1]<<endl;

return 0;

}
```

Char a[10]=”Hello”;
a[0]=H
a[1]=e
a[2]=l

++

Ex:Write a program to print each character of the string entered on a line

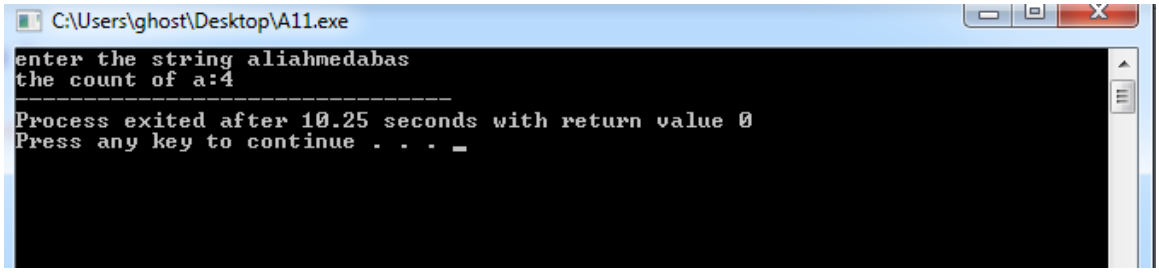
```
#include <iostream>

#include <cstring>

using namespace std;

int main()

{


string    a ;

cout<<"enter the string";

cin>>a;


for(int i=0;i<strlen(a);i++)

cout<<a[i]<<endl;

return 0;

}
```

++

Ex:Write a program to calculate how many times the letter (a) is repeated in the entered text string

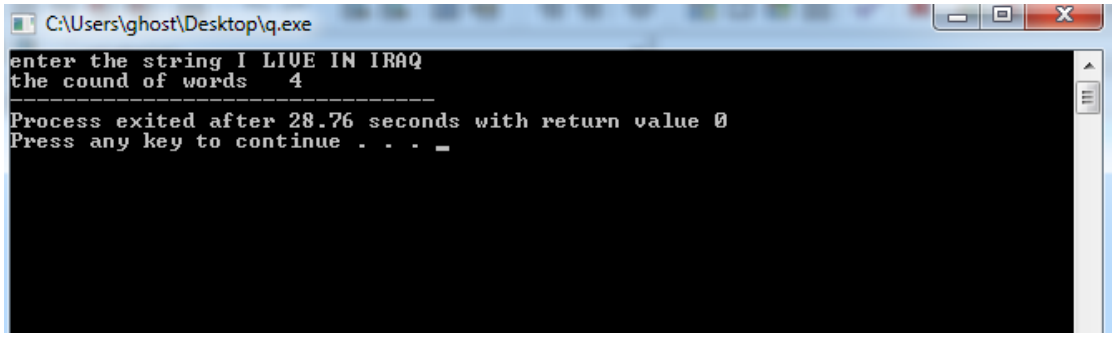
```
#include <iostream>

#include <cstring>

using namespace std;
```

```
int main()
{
int c;
char  a[10];        / or   string   a;
cout<<"enter the string";
cin>>a;

for(int i=0;i<strlen(a);i++)
if (a[i]=='a')
c++;
cout<<"count ="<<c<<endl;
return 0;
}
```




```
# include<cstring>
# include<iostream>
using namespace std;
int main()
{
        int i,c,x1=1;
        string s1;
        cout<<"enter the string ";
        getline(cin,s1);        //
```

```
        for(i=0;i<s1.length();i++)
        {
                if(s1[i]==' ')
                x1++;
        }
        cout<<"the cound of words   "<<x1;
        return 0;
}
```



++++++++++++++++++++++++++++++++++

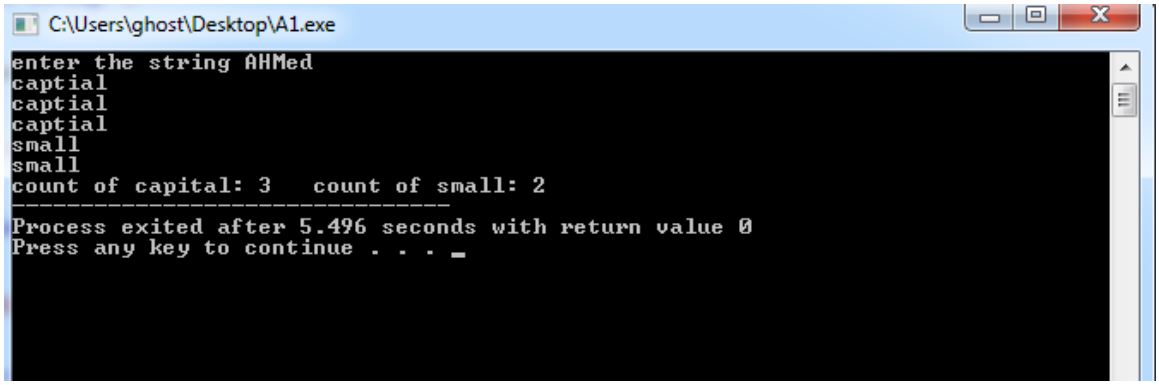
H.W: Count how many times the word (Hellow) is repeated in the text

Hellow Ali ,Hellow Ahmed, Hellow Sara and Hellow Abas

```
#include<iostream>
#include<cstring>
using namespace std;
int main()
{
        int i,c,x=0,x1=0;
        string  s;
        cout<<"enter the string:"
        cin>>s;
```

```
        for(i=0;i<strlen(s);i++){
           c= int (s[i]);
        if((c>=65)&&(c<=90))                //

        {
          cout<<"captial"<<endl;
          x++;}

          if((c>=97)&&(c<=122)){                    // or if((s[i]>='a')&&(s[i]<='z')
          cout<<"small"<<endl;
          x1++;}
    }
     cout<<"count of capital: "<<x<<"   "<<"count of small: "<<x1;
    return 0;
}
```

التنفيذ:




```
# include<iostream>
# include<cstring>
using namespace std;
int main()
{
```

```
        int i,c;  string n="AHMed";
        //cout<<"enter the string ";
        //cin>>n;
        cout<<strlen(n);
        for(i=0;i<strlen(n);i++){

        c=int (n[i]);
        if((c>=65)&&(c<=90))
          c=c+32;

          else
          if((c>=97)&&(c<=122))
          c=c-32;
        cout<<char (c)<<endl;
}
        //cout<<int('a')<<endl;
        //cout<<char(87)<<endl;
        return 0;
}
```

التنفيذ