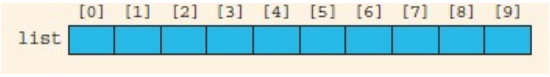
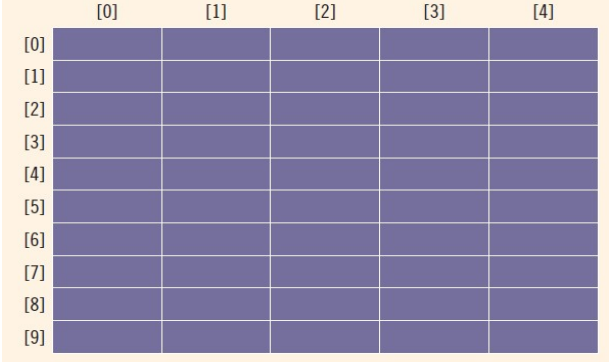
## 1.6 Array Lecture two

The data type is called **simple** if variables of that type can store only one value at a time. In contrast, in a **structured data type**, each data item is a collection of other data items. Simple data types are building blocks of structured data types. The **array** is a structured data type.

| one-dimensional array | Two-dimensional array: |
|---|---|
| dataType arrayName[intExp]; | dataType arrayName[intExp1][intExp2]; |
| int list[10]; | int list[10][5]; |
|  |  |
| list[5] = 34;// assign value | list[5][1] = 34;// assign value |
| X=list[5] +7; // it used in equation | X=list[5][2] +7; // it used in equation |
| cout<< list[5] ;// print the value | cout<< list[5][2] ;// print the value |
| for (int i=0;i<10;i++)<br>cin>>list[i];// read the values | for (int i=0;i<10;i++)<br>for (int j=0;j<5;j++)<br>cin>>list[i][j]; //read the values |
| for (int i=0;i<10;i++)<br>list[i]=I; //assign values | for (int i=0;i<10;i++)<br>for (int j=0;j<5;j++)<br>list[i][j]=1; //assign values |
| for (int i=0;i<10;i++)<br>cout<<list[i]; //print values | for (int i=0;i<10;i++)<br>for (int j=0;j<5;j++) |

| | |
|---|---|
| | cout<<list[i][j]; //print values |

**EX: C++ program to find the sum and average of an array.**

```
#include <iostream>
using namespace std;
int main()
{
        int list[10],i,sum=0;
        for (i = 0; i < 10; i++)
                cin>>list[i] ;
        for (i = 0; i < 10; i++)
                cout<<list[i]<<"  " ;
        cout<<endl;
        for (i = 0; i < 10; i++)
                sum+=list[i] ;
        cout<<"the summation of array is "<<sum;
        cout<<"\nthe average of array is "<<sum/10;
        return 0;                   }
```

**EX: C++ program to find the Largest element in the array.**

```
#include <iostream>
using namespace std;
int main()
{
        int list[10],i,large;
        for (i = 0; i < 10; i++)
                cin>>list[i] ;
        for (i = 0; i < 10; i++)
                cout<<list[i]<<"  " ;
        large=list[0];
        for (i = 1; i < 10; i++)
            if (list[i]>large)
               large=list[i];
        cout<<"\n the Largest element in the array is  "<<large;
           return 0;
  }
```

**EX: C++ program to find the Largest Element in Each Row and Each Column** in two array dimensions.

```
#include <iostream>
using namespace std;
int main()
{
        int matrix[4][4],row,col,rows,cols,largest;
        rows = 4;
        cols=4;
        for (row = 0; row < rows; row++)
                for (col = 0; col < cols; col++)
                        cin>>matrix[row][col] ;
        for (row = 0; row < rows; row++)
           {for (col = 0; col < cols; col++)
                        cout<<matrix[row][col]<<"  " ;
                 cout<<endl;}

   //Largest element in each row
        for (row = 0; row < rows; row++)
                {
                largest = matrix[row][0]; //Assume that the first element
                //of the row is the largest.
                for (col = 1; col < cols; col++)
                    if (largest < matrix[row][col])
                       largest = matrix[row][col];
                cout << "The largest element in row " << row + 1 << " = "
                << largest << endl;
                }
                //Largest element in each column
                for (col = 0; col < cols; col++)
                {
                        largest = matrix[0][col]; //Assume that the first element
                //of the column is the largest.
                        for (row = 1; row < rows; row++)
                                if (largest < matrix[row][col])
                                        largest = matrix[row][col];
                        cout << "The largest element in column " << col + 1
                        << " = " << largest << endl;              }
```

```
    return 0;}
```

**EX: C++ program replaces Main Diagonal with the Secondary Diagonal in a square array.**

```
#include <iostream>
 using namespace std;
int main()
{
        int matrix[5][5],row,col,rows,cols,a;
        rows = 5;
        cols=5;
        for (row = 0; row < rows; row++)
                for (col = 0; col < cols; col++)
                        cin>>matrix[row][col] ;
        for (row = 0; row < rows; row++)
           {for (col = 0; col < cols; col++)
                        cout<<matrix[row][col]<<"  " ;
            cout<<endl;}
        for (row = 0; row < rows; row++)
                      {a=matrix[row][row];
                       matrix[row][row]=matrix[row][rows-1-row];
                       matrix[row][rows-1-row]=a;}
         cout<<"the new array after replacing is :"<<endl;
                for (row = 0; row < rows; row++)
                        {for (col = 0; col < cols; col++)
                                cout<<matrix[row][col]<<"  " ;
                         cout<<endl;}
return 0;}
```

| | | |
|---|---|---|
| 1 | 2 | 3 |
| 4 | 5 | 6 |
| 7 | 8 | 9 |

| | | |
|---|---|---|
| 3 | 2 | 1 |
| 4 | 5 | 6 |
| 9 | 8 | 7 |

**EX: C++ program to initialize a two-dimensional array with character elements.**

```
#include <iostream>
using namespace std;
int main()
{
        char matrix[5][5],ch='a';
        int row,col,rows,cols,a;
        rows = 5;
        cols=5;
```

```
for (row = 0; row < rows; row++)
    for (col = 0; col < cols; col++)
            matrix[row][col]=ch++ ;
for (row = 0; row < rows; row++)
   {for (col = 0; col < cols; col++)
        cout<<matrix[row][col]<<"  " ;
    cout<<endl;}
return 0;}
```

| | |
|---|---|

## 1.7 C++ Strings

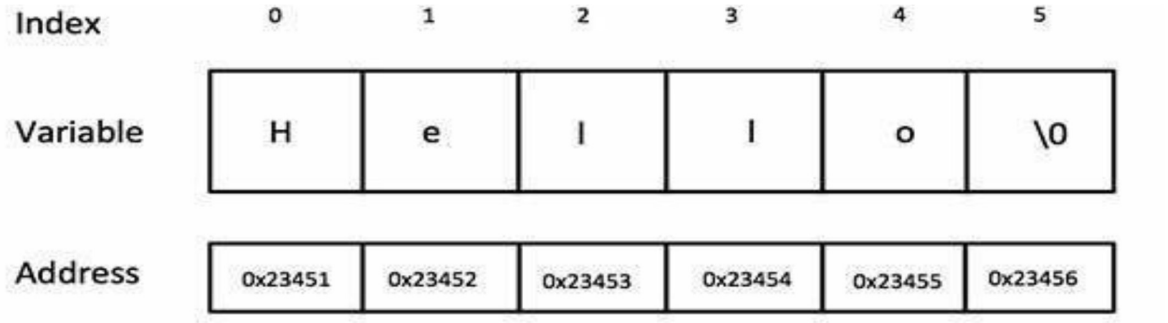
C++ provides the following two types of string representations:

- The **C-style character** string.
- The **string class type** was introduced with Standard C++.

The **C- string** originated within the C language and continues to be supported within C++. This string is a one-dimensional array of characters terminated by a **null** character '\0'.

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

char greeting[] = "Hello";




**EX: c++ program to initialize and print null-terminated string**

```
#include <iostream>
using namespace std;
int main () {
char greeting[6] = {'H', 'e', 'l', 'l', 'o', '\0'};
cout << "Greeting message: ";
cout << greeting << endl;
return 0;
```

}

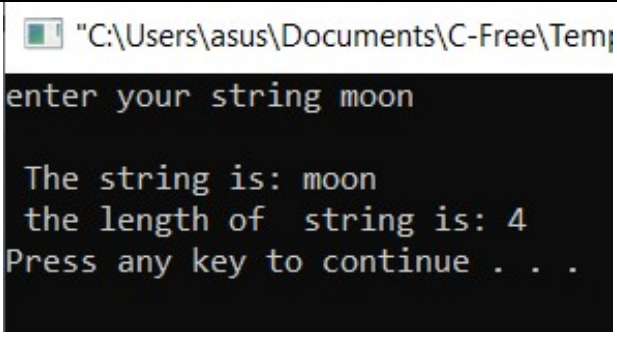
C++ supports a wide range of functions that manipulate **null-terminated strings** :

| **strcpy(s1,s2);** | **Copies string s2 into string s1.** | **strcat(s1, s2);** | **Concatenates string s2 onto the end of string s1.** |
|---|---|---|---|

```
#include <iostream>
using namespace std;
int main () {
  char st1[10],st2[10];
  cout << "enter your strings ";
  cin>>st1>>st2;
  cout<<"\n first string is: "<<st1;
  cout<<"\n second string is: "<<st2;
  strcpy(st1,st2);
  cout<<"\n first string is: "<<st1;
  cout<<"\n second string is: "<<st2;
 return 0;
}
```

```
#include <iostream>
using namespace std;
int main () {
 char st1[10],st2[10];
 cout << "enter your strings ";
 cin>>st1>>st2;
 cout<<"\n first string is: "<<st1;
 cout<<"\n second string is: "<<st2;
 strcat(st1,st2);
 cout<<"\n concatenating string is : "<<st1;
 return 0;
}
```

| **strlen(s1);** | Returns the length of string s1. |
|---|---|




```
#include <iostream>
using namespace std;
int main () {
char st1[10];
cout << "enter your string ";
cin>>st1;
cout<<"\n The string is: "<<st1;
cout<<"\n the length of  string is:
"<<strlen(st1)<<endl;
return 0;
}
```

EX: C++ program used to manipulate **null-terminated strings.**

```
#include <iostream>
#include <cstring>
using namespace std;
int main () {
char str1[10] = "Hello";
char str2[10] = "World";
int len ;
```

```
// concatenates str1 and str2
strcat( str1, str2);
cout << "strcat( str1, str2): " << str1 << endl;
// total lenghth of str1 after concatenation
len = strlen(str1);
cout << "strlen(str1) : " << len << endl;
return 0;
}
```

**String class** is part of the C++ library that supports much functionality over C-strings.

string st;

st=" Hello"

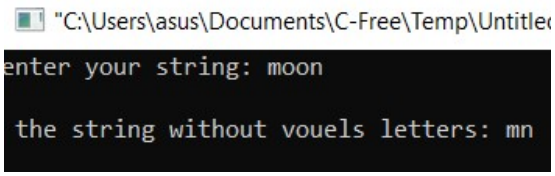
| St[] | H | e | l | l | 0 |
|---|---|---|---|---|---|

| index | 0 | 1 | 2 | 3 | 4 |
|---|---|---|---|---|---|

The data type string contains several other functions for string manipulation. The the following table describes some of these functions: suppose strVar ="C++" and str =" language"

| Expression | Effect | EX |
|---|---|---|
| strVar[index] | Returns the element at the position specified by index. | cout<<strVar[2];<br>Output: + |
| strVar.append(str) | Appends str to strVar. | strVar.append(str);<br>Output:C++ language |
| strVar.append(n, ch) | Appends n copies of ch to strVar, in which ch is a char variable or a char constant. | strVar.append(2, '&');<br>Output:c++&& |
| strVar.compare(str) | Returns 1 if strVar <> str; returns 0 if strVar == str | cout<<strVar.compare(strVar);<br>Output:0<br><br>cout<<strVar.compare(str);<br>Output:1 |
| strVar.length() | Giving the number of characters strVar. | cout<< strVar.length();<br>Output:3 |
| strVar.swap(str1); | Swaps the contents of strVar and str1. str1 is a string variable. | strVar.swap(str);<br>cout<<strVar<<" "<<str<<endl;<br><br>Output:language c++ |

**EX:c++ program that prompts the user to input a string and remove all the vowels from the string. For example, if str = "There," after removing all the vowels, str = "Thr."**




```
#include <iostream>
#include <string>
using namespace std;
int main () {
string st;
cout<<"enter your string: ";
cin>>st;
int i=0;
while(i<st.length())
   if (st[i]=='A'||st[i]=='E'||st[i]=='I'||st[i]==O'||st[i]=='U'||st[i]=='Y'||
      st[i]=='a'||st[i]=='e'||st[i]== 'i'||st[i]== 'o'||st[i]=='u'||st[i]=='y')
              for(int j=i;j<st.length();j++)
                   st[j]=st[j+1];
   else
         i++;
cout<<"\n the string without vowel letters: "<<st;
return 1;}
```

**EX: c++ program detect if the string can be read from both sides:**

```
#include <iostream>
#include <string>
using namespace std;
int main () {
        string st1,st2="";
        cout<<"enter your string: ";
        cin>>st1;
        int len=st1.length();
        for(int i=len-1;i>=0;i--)
           st2.append(1, st1[i]);
        cout<< "\n the swapping string is "<<st2;
        if (!st1.compare(st2))
           cout<<"\n string read from both sides\n";
        else
```