Programming fundamentals 2
Chapter 1:Array

Miss: Hanan Hardan

Arrays in C++

- Data types are of two kinds:
  - simple data types (e.g. int, float, double, char)
  - Structured data type: (e.g. arrays)

- An **array** is a collection of two or more adjacent memory cells, called array elements, that are associated with a particular symbolic name.

- In C++ each array has: name, data type, size

- Several operations are allowed on the array: Read, Write, Search, Sum, Min, Max, Sort, etc..

- Arrays are of two kinds:
  - Arrays of one-dimension
  - Arrays of two-dimension

One-Dimensional Arrays

- **Declaration of one-dimension array**

Syntax: atype aname [size] ; // uninitialized array

atype aname [size] = { initialization list } ;

where

atype is any data type;

aname is the name given to the array;

size represents the number of elements in the array.

initialization list is the list of initial values given to the array.

Declaration of Arrays

For example ,

```
int  x [ 3 ];
```

- This tells the compiler to associate 3 memory cells with name x.
- These cells will be adjacent to each other in memory.
- Each element of array x contains a value of integer type

- More than one array can be declared on a line

```
int age [10] , height [10] , names [20] ;
```

- Mix declaration of variables with declaration of arrays

```
int i , j , age [10] ;
```

Initializing an Array

Example1:

```
int  Y [ 4 ] = { 2, 4, 6, 8 } ;
```

This initializes array Y to have 4 elements which contain 2, 4, 6, 8.

2	4	6	8

Example2:

```
int age [ 10 ] = { 0,0,0,0,0,0,0,0,0,0 } ;
```

Example3:

```
int age[ 10 ] = { 0 } ;
```

0	0	0	0	0	0	0	0	0	0

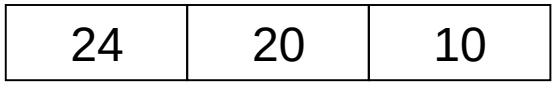
Example4:

```
int age [ ] = { 1,2,3,4,5,6,7,8,9,10 } ;
```

Accessing One-Dimensional Array

int x [3]={24,20,10} ;

Array x in memory:



How to process the data stored in an array?

Syntax:

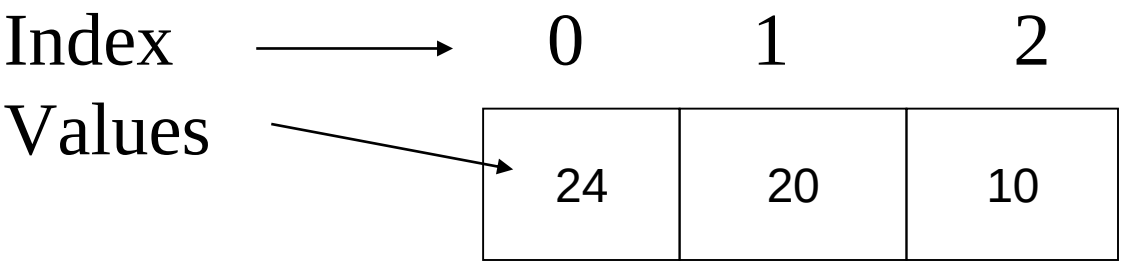
aname [**index**]

- **index** is the subscript that is used to reference the desired element.

The array indexing starts from 0 until the fixed size -1.

Accessing One-Dimensional Array

Array **x** in memory:



Accessing the array:

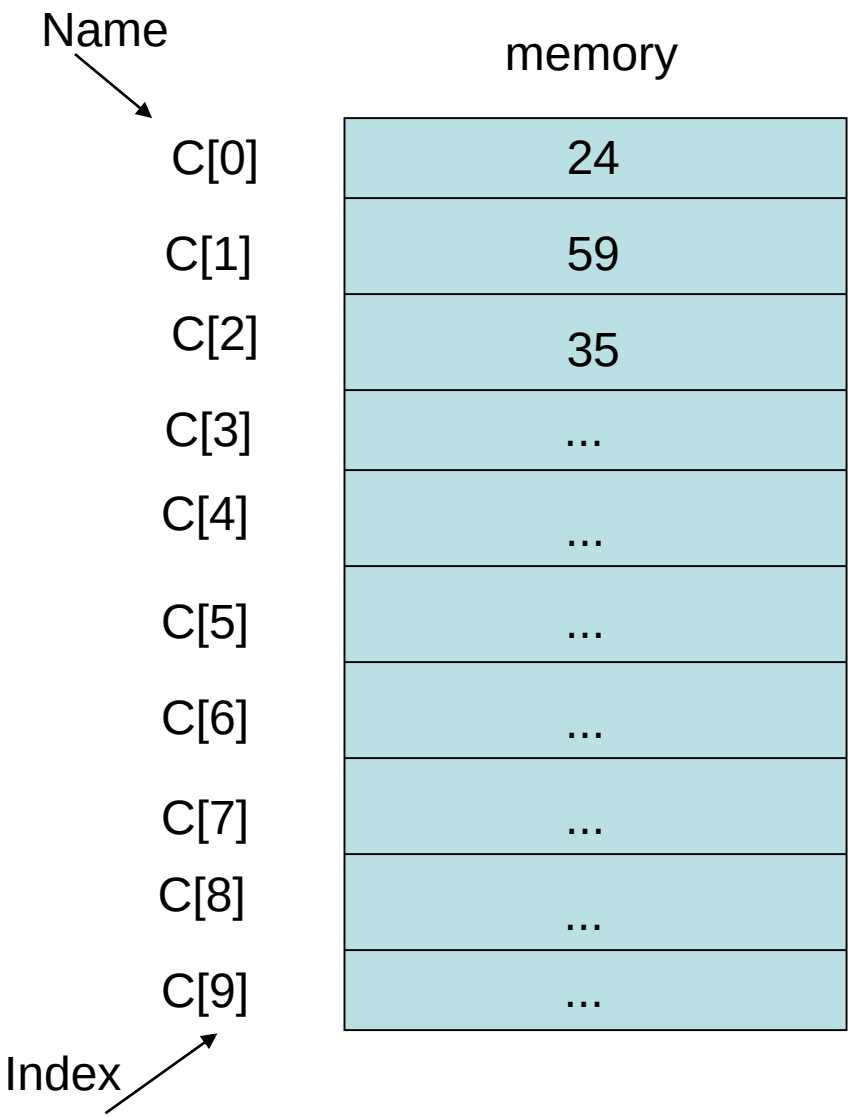
x [0] to access the first element of x

x [1] to access the second element of x

x [2] to access the third element of x

Storage of an array in memory

Example:
int C[10];
C[0]=24;
C[1]=59;
C[2]=35;



- **Example:** Write C++ program that output the array elements. Where L[5]={1,2,3,4,5};

```
#include <iostream>
using namespace std;
void main ()
{
 int L [5]={1,2,3,4,5};
 for( int i=0;i<5;i++)
     {
       cout<<L[i] ;
     }
}
```

- **Example:** Write C++ program that read array of size 10 integer numbers.

```
#include <iostream>
using namespace std;
void main ()
{
 int L [10];
cout<<"please enter 10 integer number";
for(int i=0;i<10;i++)
    {
      cin  >> L[i] ;
    }
}
```

Examples on One-Dimensional Arrays

• **Example 1:** Write a C++ program that stores the first 5 integers that are multiples of 5 into array A and reads data into array B;computes the sum of the two arrays and stores the result in array C.

```
# include <iostream.h>
void main ( )
{ int  A [5] ;    //declaring array A of 10 integers
  int B [5] , C [5];       //declaring arrays B and C of 10 integers
  for ( int i = 0; i <5  ; i++ )
     { A[ i ] = ( i + 1 ) * 5;
       cout << "enter new element in array B";
       cin >> B[ i ] ;
       C [ i ]  = A[ i ] + B [ i ] ;
       cout  <<  C [ i ] << "    "  ;
      }  }
```

Example1..Cont.

The trace of the previous example shows the arrays as follows if B elements are 7 6 10 13 23:

A

0	1	2	3	4
5	10	15	20	25

B

0	1	2	3	4
7	6	10	13	23

C

0	1	2	3	4
12	16	25	33	48

Copying Arrays

- **Data types should be identical**
- **Size should be same**

```
int a [ 10 ] ={7,1,3,5,8,9,1,2,8,4};
int b [ 10 ] ;
 for ( i =0 ; i < 10 ; i ++ )
      b [ i ] = a [ i ] ;
```

Example 2

Example 2:

Write a C++ program to read 10 integers and store them in array A. Then it finds the even numbers to store them in array B and the odd numbers to store them in array C.

Example 2 .. Cont.

```
#include <iostream.h>
void main ( )
{ int i;
  int A [10], B[10], C[10] ;
  cout << " Enter 10 integers : " ;
   for (i=0 ; i <10; i++)
   {  cin >>  A[i] ;
      if (A[i] % 2 == 0)
         B[i] = A[i] ;
      else
         C[i] = A[i];
    }
    cout << "B element = "  << "        C element = "  << endl;
    for (i=0; i<10; i++)
    { cout << B[i] << "          "  << C[i] << endl ;
    }
}
```

Examples on One-Dimensional Arrays

Example 3:

The resultant arrays B and C in example 2 contain data stored in non consecutive locations in the arrays. Modify the program of example 2 so that the data are stored in consecutive locations.

```
#include <iostream.h>
void main ( )
{ int i, j = 0 , k = 0 ;
   int A [10], B[10], C[10] ;
  for ( i= 0 ; i <10; i++)
   {  cin >>  A[i] ;
      if (A[i] % 2 == 0)
        {  B [ j ] = A [ i ] ;
            j  ++ ;          }
      else
        {  C [k] = A [ i ];
            k ++ ;
         }   }
  cout << "B element = " ;
  for (i=0; i<j; i++)
         cout << B[i] <<"    " ;
  cout<<endl;
 cout << "C element = " ;
 for (i=0; i<k; i++)
   cout << C[i] << "      " ;
}
```

Example 4

The problem:

Write C++ program that searches for an integer in array of 10 integers. A proper message should be printed out.

The Analysis:

A given array of integer numbers is going to be searched in order to find a given number.

Requirements:

Input: an integer number n, an array A of 10 integers

Output: a message “yes found” or “no not found” according to weather the number is found or not.

The C++ Program

```
#include <iostream.h>
void main ( )
{
int  L [10] ;
int CurElem, SearchVal;
bool Found;
// ----------------------- Reading L ----------------------------
cout<<"Enter 10 integers:";
CurElem = 0 ;
while (CurElem < 10 )
{
    cin>> L[CurElem] ;
    CurElem = CurElem + 1;
}
//---------------------------- End Reading L -----------------
```

```
cout<< "Enter the number that you ant to search for: ";
cin>>SearchVal;
Found = false;
CurElem = 0;
while (CurElem < 10 && !Found )
{
  if  (L[CurElem] == SearchVal)
        Found = true;
  else
        CurElem =CurElem + 1;
}
 if ( Found == true)
          cout<<" Yes, the number is found ";
 else
         cout<<" No, the number is not found" ;
}
```

Arrays of Two-Dimensions

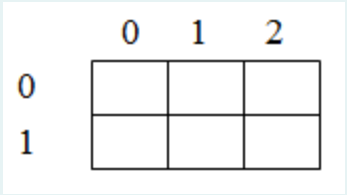
Syntax (in C++):

atype aname [nrows] [ncolumns] ;

Example:

int B [2] [3] ;

Array in diagram:



Arrays of Two-Dimensions

- **Reading array of two-dimensions:**

Syntax (in C++):

```
int   B [2] [3] ;
int  i , j ;
for (  i = 0 ; i < 2 ; i++ )
     for (  j = 0 ; j < 3 ; j++ )
            cin >> B[ i ] [ j ] ;
```

After first outer loop

Input

[0]	5	2	9

After second outer loop

Input

[0]	5	2	9
[1]	7	0	4

Example1

Write C++ program that reads array A of size (2 x 3) and finds the sum of the elements in each row.

```
#include <iostream>
using namespace std;
void main ( )
{  int i, j , Rsum = 0 ;
   int B[2][3];
   cout << "Enter  6 array elements: " ;
   for ( i = 0 ; i < 2 ; i++ )
       for ( j = 0 ; j < 3 ; j++)
                cin >> B[i][j] ;
// Process the array now
   for ( i = 0 ; i < 2 ; i++ )
      {
       for ( j = 0 ; j < 3 ; j++)
             {   Rsum = Rsum + B[i][j] };
        cout <<  " sum of row no. " <<i<< " is  " << Rsum<<endl;
        Rsum = 0;
     }
}
```

Example2 of Two-Dimensional Array

• Write algorithm and a C++ program that reads an array of size (3 x 3) and finds the product of the diagonal elements.