Arrays in C++

* What is an array?
  - An array is a collection of logically related data items of the same datatype, that share a common name.
  - All the items of array are stored in contiguous memory locations.
  - Individual elements of an array are accessed and manipulated using the array name followed by their index.
  - It can be used to represent
    - ⇒ a vector ⇒ one dimensional array
    - ⇒ matrix ⇒ two dimensional array.
  - Can be used for storing and manipulating strings (sequence of characters)

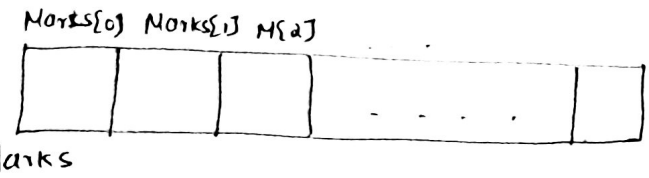
* Array Definition
  - The array variable must be defined before its use.
  - Syntax for array definition is

    datatype Arrayname [size];

    - ⇒ datatype ⇒ primitive or user defined.
    - ⇒ size ⇒ maximum number of elements the array can hold.

  (Eg) int marks[50]; ⇒ integer array of size 50.
  float price[50];
  char name[30];

  - The definition tells the compiler that marks is an array of type integer and can store 50 integers.
  - Compiler reserves 4 bytes of memory for each integer array element.



* Array initialization at Definition

- Arrays can be initialized at the point of their definition

(Eg) int a[5] = { 10, 20, 30, 40, 50 } ;

(or)

int a[] = { 10, 20, 30, 40, 50 } ;

→ If the array size is omitted, the C++ compiler automatically allocates memory space for all elements in the array.

- Other egs :

char xyz[10] = { 'a', 'b', 'c' } ;

→ Here the no. of elements is less than the size

→ The values for 4th to 10th elements of the array will be set to zero automatically.

* Accessing Array Elements

- Once an array variable is defined, its elements can be accessed using an index (or subscript) in bracket [ ].

Syntax:

```
Arrayname [index] ;
```

index → integer constant
→ variable
→ expression

- In an array of N elements, the first element is indexed by zero and the last element by N-1

(Eg) int a[5] = { 10, 20, 30, 40, 50 } ;

a[0] refers to 1st element (ie) 10.
a[4] refers to last element (ie) 50.

Note:

* The amount of storage required by the array is calculated using the type and size of array.
  - Size is calculated in bytes.
  - Total bytes = size of (data type) * size of array.

    size of char [20] is 20 bytes.
    int [20] is 40 (or) 80 bytes.

* No array bound validation
  - C++ does not support bound checking
  → does not check for array index validity while accessing array elements.

(Eg)
```
void main()
{
  int a[20];
  a[35] = 1250;
  cout << a[35];
}
```
// No compiler nor runtime error will be displayed.

1) Program to sort n numbers in ascending order.

```
#include <iostream.h>
void main()
{
  int a[10], n, temp;
  cout << "Enter the no. of elements" << endl;
  cin >> n;
  cout << "Enter the elements" << endl;
  for(int i=0; i<n; i++)
    cin >> a[i];
```

```
for (int i=0 ; i<n-1; i++)
  {
   for (int j=i+1; j<n ; j++)
    {
      if (a[i] > a[j])
        {
          temp= a[i];
          a[i] = a[j];
          a[j] = temp;
        }
    }
  }
cout << " The elements in ascending order is" <<endl;
for (int i = 0 ; i<n ; i++)
    cout << a[i] ;
}
```

o/p: Enter the no. of elements
5
Enter the elements
2 3 5 1 4
The elements in ascending order is
1 2 3 4 5

2) Program to find the smallest and biggest element in an array.

```
void main()
 {
  int  a[10], n , small, big
  cout << "Enter no. of elements" << endl;
  cin >> n;
  cout << "Enter the elements" <<endl;
  for (int i=0; i< n; i++)
      cin >> a[i];
  small = a[0];
  big = a[0];
  for (i=1; i<n ; i++)
    {
      if a[i] < small
          small = a[i];
      else
        if a[i] > big
    };     big = a[i];
```

```
cout << "The biggest no is" <<big
cout << " The smallest no. is"
                << small;
```

Multi-dimensional Arrays

* Definition

Syntax:

```
datatype arrayname [size1][size2] ... [sizen];
```

(Eg) int abc[4][3][3]. ⇒ 3 dimensional array with array name abc.

Two Dimensional Arrays:

* Definition:

Syntax:

```
datatype arrayname [size1] [size2];
```

size1 → row size
size2 → column size

(Eg) int a[3][4];
float b[3][2];

* 2D array initialization at definition

2D array can be initialized using the syntax:

Syntax:

```
datatype matrixname [rowsize][col_size] = { {1st row elements},
                                            {2nd row elements}, ... {n-1 th row elements}};
```

(Eg) int a[3][4] = {{1,2,3,4}, {5,6,7,8}, {9,10,11,12}};

Note: (i) The first subscript (size of row) can be omitted

(Eg) int a[][2] = {{1,2}, {3,4}, {5,6}, {7,8}}

(ii) The inner braces can also be omitted. But it lacks readability

(Eg) int a[][2] = {1,2,3,4,5,6,7,8};

* Accessing 2D array elements :

- 2D array elements can be accessed using the syntax,

Syntax

arrayname [i] [j] ;

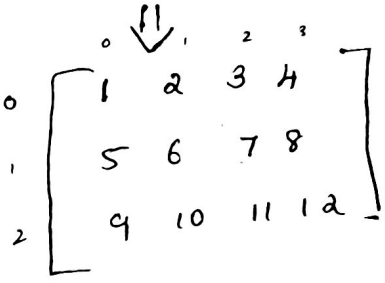
i → row number

j → column number .

(Eg) int a [3][4] = { {1,2,3,4} , {5,6,7,8} , {9,10,11,12} }

a [2][0] ⇒ refers to 9

a [1][2] ⇒ refers to 7 . . .



1) Program to add two matrices .

```
#include <iostream.h>
void main()
{
  int a[10][10] , b[10][10] , c[10][10] ;
  int x , y , p , q , i , j ;
  cout << " Enter the no. of rows and columns of Matrix A \n" ;
  cin >> x >> y ;
  cout << " Enter the no. of rows and columns of Matrix B \n" ;
  cin >> p >> q ;
  if ((x == p) && (y == q))
  {
    cout << "Enter Matrix A elements \n" ;
    for( i=0 ; i < x ; i++)
      for( j=0 ; j< y ; j++)
        cin >> a[i][j] ;
```

```
cout << "Enter matrix B elements \n";
    for (i=0; i<p; i++)
        for(j=0; j<q; j++)
            cin >> b[i][j];
for(i=0; i<x; i++)
    for(j=0; j<y; j++)
        c[i][j] = a[i][j] + b[i][j];
cout << "The resultant matrix after addition\n";
    for(i=0; i<x; i++)
        for(j=0; j<y; j++)
            cout << c[i][j] << " ";
        cout << endl;
    }
}
```

O/P:

Enter the no. of rows and columns of Matrix A

3 4

Enter the no. of rows and columns of Matrix B

3 4.

Enter Matrix A elements

$$\begin{bmatrix} 1 & 2 & 3 & 4 \\ 5 & 6 & 7 & 8 \\ 9 & 10 & 11 & 12 \end{bmatrix}$$

Enter Matrix B elements

$$\begin{bmatrix} 1 & 2 & 3 & 4 \\ 5 & 6 & 7 & 8 \\ 9 & 10 & 11 & 12 \end{bmatrix}$$

The resultant matrix after addition

$$\begin{bmatrix} 2 & 4 & 6 & 8 \\ 10 & 12 & 14 & 16 \\ 18 & 20 & 22 & 24 \end{bmatrix}$$

2) Program for Matrix Multiplication

```
#include<iostream.h>
Void main()
{
int a[10][10], b[10][10], c[10][10], x, y, p, q;
cout<<"Enter the order of matrix A\n";
cin>>x>>y;
cout<<"Enter the order of matrix B\n";
cin>>p>>q;
if (y==p)
 {
 cout<<"Enter the matrix A"<<endl;
 for(int i=0; i<x; i++)
  for(int j=0; j<y; j++)
    cin>>a[i][j];
cout<<"Enter the matrix B\n";
 for(int i=0; i<p; i++)
  for(int j=0; j<q; j++)
     cin>>b[i][j];
for(int i=0; i<x; i++)
  for(int j=0; j<q; j++)
    {
     c[i][j]=0;
     for(int k=0; k<p; k++)
       c[i][j]=c[i][j]+a[i][k]*b[k][j];
    }
cout<<"The resultant matrix after multiplication\n";
 for(int i=0; i<x; i++)
  for(int j=0; j<q; j++)
     cout<<c[i][j]<<" ";
} cout<<endl; }
```

Arrays within a class

- Arrays can be used as member variables in a class.

(Eg)

```
#include <iostream.h>
const int size=5;
class student
{
 int rollno;
 int marks[size];
 public:
  void getmarks();
  void total();
};
void student :: getmarks()
 {
 cout << "Enter roll no : \n";
 cin>> rollno;
 for(int i=0; i<size; i++)
  {
  cout << "Enter marks in subject " << (i+1) << endl;
  cin >> marks[i];
  }
 }
void student :: total()
 {
 int total=0;
 for(int i=0; i<size; i++)
  {
  total += marks[i];
  }
 cout << "Total marks \n" << total;
 }
```

```
void main( )
{
 Student s1 , s2 ;     // Creates obj for 2 students
 s1. getmarks( ) ;     // Get marks in 5 subjects for 1st student
 s1 . total ( ) ;      // Find total marks for 1st student
 s2 . getmarks( ) ;    // Get marks for 2nd student
 s2 . total ( ) ;      // Find total marks for 2nd student.
}
```

- Here the memory space for array is allocated when object of class is declared.

Array of objects

* Array of variables of type class.

- For the above program, an array of objects can be created as follows, each element representing individual student.

```
void main()
 {
  student s[5] ;
  for(int i=0 ; i<5 ; i++)
   {
   cout<< "\n Enter details of student " << (i+1) << endl ;
    s[i]. getmarks( ) ;
    }
  for(int i=0 ; i<5 ; i++ )
   {
   cout<< " Total marks for student " << (i+1) << endl ;
    s[i]. total ( ) ;
    }
 }
```