Tassadaq Hussain

The main purpose of C++ programming is to add object orientation to the C programming language and classes are the central feature of C++ that supports object-oriented programming and are often called user-defined types.

A class is used to specify the form of an object and it combines data representation and methods for manipulating that data into one neat package. The data and functions within a class are called members of the class.

Class

A class definition starts with the keyword class followed by the class name; and the class body, enclosed by a pair of curly braces. A class definition must be followed either by a semicolon or a list of declarations. For example, we defined the Box data type using the keyword class as follows:

```
#ifndef DATE_H
#define DATE_H
class Date
{
private:
    int m_nMonth;
    int m_nDay;
    int m_nYear;
    Date() { } // private default
constructor
 public:
    Date(int nMonth, int nDay, int
nYear);
    void SetDate(int nMonth, int nDay,
int nYear);
    int GetMonth() { return m_nMonth; }
    int GetDay()   { return m_nDay; }
    int GetYear() { return m_nYear; }
};
#endif
```

```
#include "Date.h"
// Date constructor
Date::Date(int nMonth, int nDay,
    int nYear)
{
     SetDate(nMonth, nDay, nYear);
}
// Date member function
void Date::SetDate(int nMonth, int
    nDay, int nYear)
{     m_nMonth = nMonth;
     m_nDay = nDay;
     m_nYear = nYear;
}
```

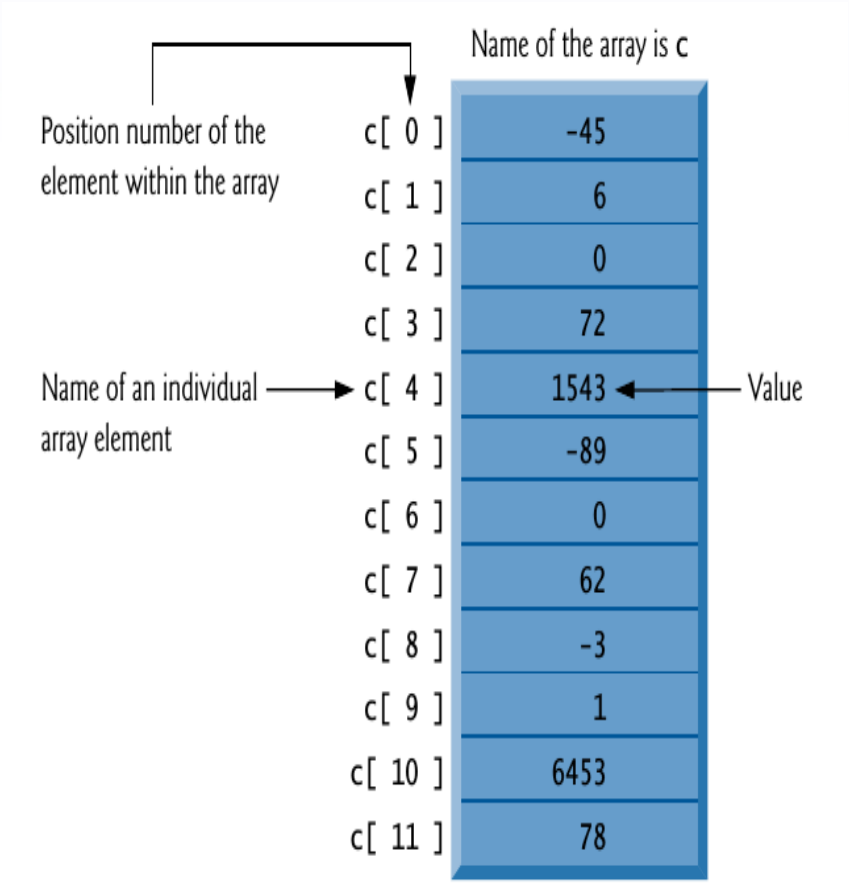
Memory Allocation

```
Int a;
Char b;
int *my_var;
my_var     = (int *)malloc(size_my_var * sizeof (int));
```

Arrays and Vectors

Arrays are data structures consisting of related data items of the same type.

Array of 12 Elements C[12]



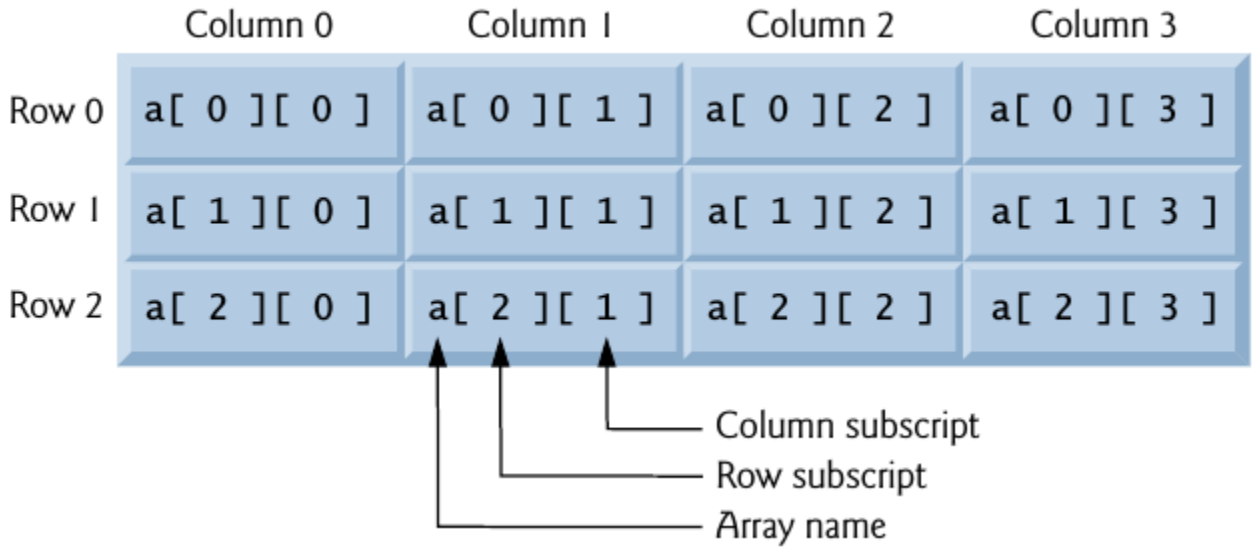

```
int c[12] = { -45 ,6, 0, 72, 1543,
    -89, 0, 62, -3, 1, 6453, 78};
```

Example Using an Array

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

int main()
{
   int n[ 10 ]; // n is an array of 10 integers

   // initialize elements of array n to 0
   for ( int i = 0; i < 10; ++i )
      n[ i ] = 0; // set element at location i to 0

   cout << "Element" << setw( 13 ) << "Value" << endl;

   // output each array element's value
   for ( int j = 0; j < 10; ++j )
      cout << setw( 7 ) << j << setw( 13 ) << n[ j ] << endl;
} // end main
```

Multidimensional Arrays

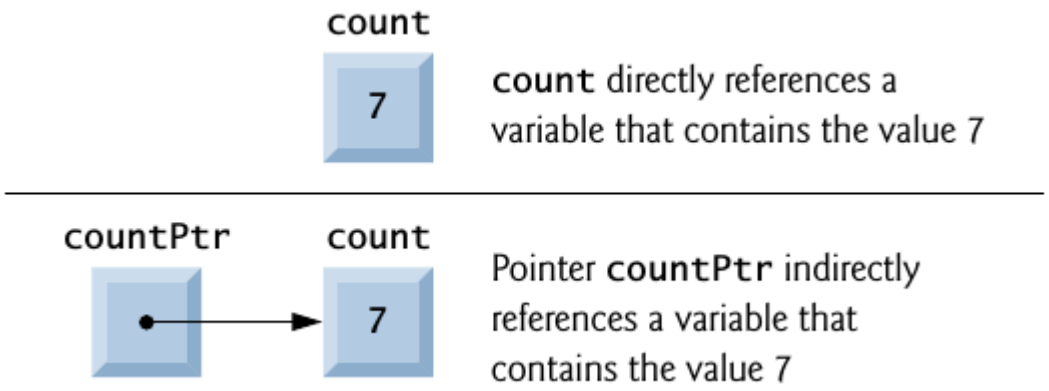
Int name_array[row][column]




```
int array1[ rows ][ columns ] = { { 1, 2, 3 }, { 4, 5, 6 } };
```

Pointer

Pointer variables contain memory addresses as their values. contains a specific value. A pointer contains the memory add



int y = 5; // declare variable y
int *yPtr; // declare pointer variable yPtr

```
yPtr = &y; // assign address of y to yPtr
```