Pointers and Arrays

PRESENTED BY-
NEELAM SINGH
DEPARTMENT OF COMPUTER SCIENCE

INTRODUCTION

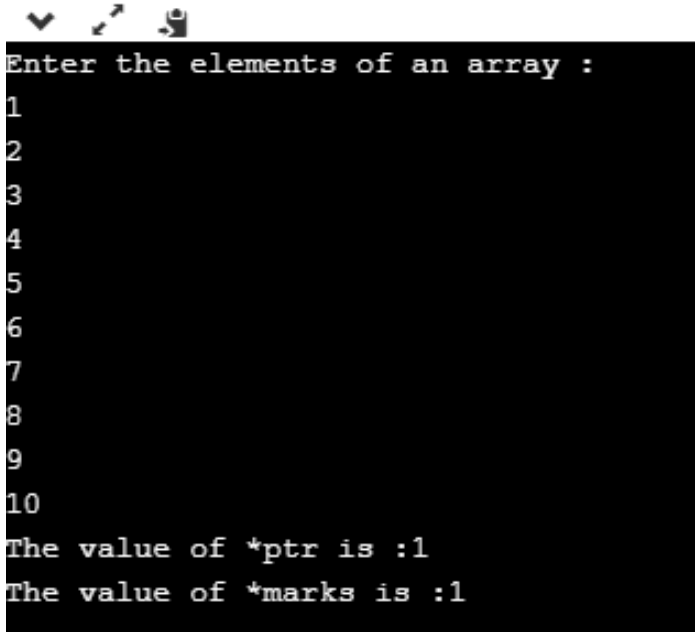
- Array and pointers are closely related to each other. In C++, the name of an array is considered às a pointer, i.e., the name of an array contains the address of an element. C++ considers the array name as the address of the first element. For example, if we create an array, i.e., marks which hold the 20 values of integer type, then marks will contain the address of first element, i.e., marks[0]. Therefore, we can say that array name (marks) is a pointer which is holding the address of the first element of an array.

Let's understand this scenario through an example.

```
#include <iostream>
using namespace std;
int main()
{
    int *ptr;  // integer pointer declaration
    int marks[10]; // marks array declaration
    std::cout << "Enter the elements of an array :" << std::endl;
    for(int i=0;i<10;i++)
    {
        cin>>marks[i];
    }
    ptr=marks; // both marks and ptr pointing to the same element..
    std::cout << "The value of *ptr is :" <<*ptr<< std::endl;
    std::cout << "The value of *marks is :" <<*marks<<std::endl;
}
```

OUTPUT

- In the above code, we declare an integer pointer and an array of integer type. We assign the address of marks to the ptr by using the statement ptr=marks; it means that both the variables 'marks' and 'ptr' point to the same element, i.e., marks[0]. When we try to print the values of *ptr and *marks, then it comes out to be same. Hence, it is proved that the array name stores the address of the first element of an array.



Array of Pointers

- An array of pointers is an array that consists of variables of pointer type, which means that the variable is a pointer addressing to some other element. Suppose we create an array of pointer holding 5 integer pointers; then its declaration would look like:

```
int *ptr[5];        // array of 5 integer pointer.
```

- In the above declaration, we declare an array of pointer named as ptr, and it allocates 5 integer pointers in memory.

- The element of an array of a pointer can also be initialized by assigning the address of some other element. Let's observe this case through an example.

int a; // variable declaration. ptr[2] = &a;

In the above code, we are assigning the address of 'a' variable to the third element of an array 'ptr'.

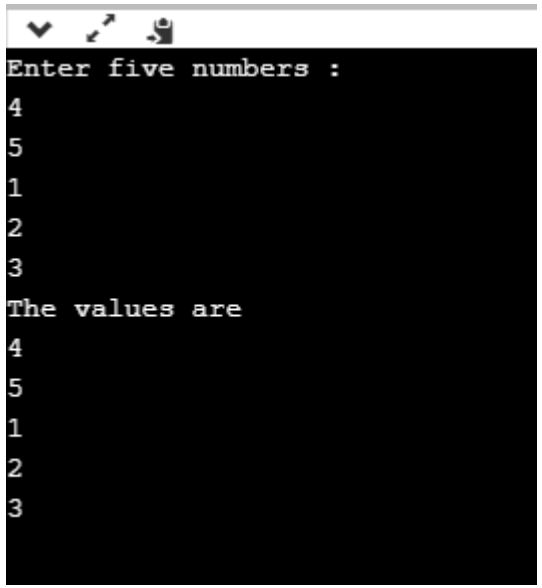
- We can also retrieve the value of 'a' be dereferencing the pointer.

*ptr[2];

Let's understand through an example.

```
#include <iostream>
using namespace std;
int main()
{
    int ptr1[5]; // integer array declaration
   int *ptr2[5]; // integer array of pointer declaration
  std::cout << "Enter five numbers :" << std::endl;
for(int i=0;i<5;i++)
 {
  std::cin >> ptr1[i];
    }
    for(int i=0;i<5;i++)
    {
        ptr2[i]=&ptr1[i];
    }
    // printing the values of ptr1 array
    std::cout << "The values are" << std::endl;
    for(int i=0;i<5;i++)
    {
        std::cout << *ptr2[i] << std::endl;
    }
```

- In the above code, we declare an array of integer type and an array of integer pointers. We have defined the 'for' loop, which iterates through the elements of an array 'ptr1', and on each iteration, the address of element of ptr1 at index 'i' gets stored in the ptr2 at index 'i'.
- **Output**



THANK YOU