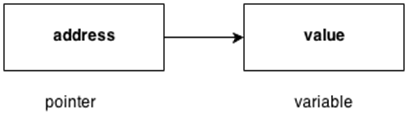
**C++ Pointers**

The pointer in C++ language is a variable, it is also known as locator or indicator that points to an address of a value.



**Advantage of pointer**

1) Pointer reduces the code and improves the performance, it is used to retrieving strings, trees etc. and used with arrays, structures and functions.

2) We can return multiple values from function using pointer.

3) It makes you able to access any memory location in the computer's memory.

**Usage of pointer**

There are many usage of pointers in C++ language.

**1) Dynamic memory allocation**

In c language, we can dynamically allocate memory using malloc() and calloc() functions where pointer is used.

**2) Arrays, Functions and Structures**

Pointers in c language are widely used in arrays, functions and structures. It reduces the code and improves the performance.

Symbols used in pointer

|  |  |  |
| --- | --- | --- |
| **Symbol** | **Name** | **Description** |
| & (ampersand sign) | Address operator | Determine the address of a variable. |
| ∗ (asterisk sign) | Indirection operator | Access the value of an address. |

Declaring a pointer

The pointer in C++ language can be declared using ∗ (asterisk symbol).

1. **int** ∗   a; //pointer to int
2. **char** ∗  c; //pointer to char

Pointer Example

Let's see the simple example of using pointers printing the address and value.

1. #include <iostream>
2. **using** **namespace** std;
3. **int** main()
4. {
5. **int** number=30;
6. **int** ∗   p;
7. p=&number;//stores the address of number variable
8. cout<<"Address of number variable is:"<<&number<<endl;
9. cout<<"Address of p variable is:"<<p<<endl;
10. cout<<"Value of p variable is:"<<\*p<<endl;
11. **return** 0;
12. }

Output

Address of number variable is:0x7ffccc8724c4

Address of p variable is:0x7ffccc8724c4

Value of p variable is:30

Pointer Program to swap 2 numbers without using 3rd variable

1. #include <iostream>
2. **using** **namespace** std;
3. **int** main()
4. {
5. **int** a=20,b=10,∗p1=&a,∗p2=&b;
6. cout<<"Before swap: ∗p1="<<∗p1<<" ∗p2="<<∗p2<<endl;
7. ∗p1=∗p1+∗p2;
8. ∗p2=∗p1-∗p2;
9. ∗p1=∗p1-∗p2;
10. cout<<"After swap: ∗p1="<<∗p1<<" ∗p2="<<∗p2<<endl;
11. **return** 0;
12. }

Output

Before swap: ∗p1=20 ∗p2=10

After swap: ∗p1=10 ∗p2=20