

Pointers in C++

Pointer is a variable in C++ that holds the address of another variable. They have data type just like variables, for example an integer type pointer can hold the address of an integer variable and a character type pointer can hold the address of char variable.

Syntax of pointer

```
data_type *pointer_name;
```

How to declare a pointer?

```
/* This pointer p can hold the address of an integer
 * variable, here p is a pointer and var is just a
 * simple integer variable
 */
int *p, var
```

Assignment

As mentioned above, an integer type pointer can hold the address of another int variable. Here we have an integer variable var and pointer p holds the address of var. To assign the address of variable to pointer we use **ampersand symbol (&)**.

```
/* This is how you assign the address of another variable
 * to the pointer
 */
p = &var;
```

How to use it?

```
// This will print the address of variable var
cout<<&var;

/* This will also print the address of variable
 * var because the pointer p holds the address of var
 */
cout<<p;

/* This will print the value of var, This is
 * important, this is how we access the value of
 * variable through pointer
 */
cout<<*p;
```

Example of Pointer

Lets take a simple example to understand what we discussed above.

```
#include <iostream>
using namespace std;
int main(){
    //Pointer declaration
    int *p, var=101;

    //Assignment
    p = &var;
```

```

    cout<<"Address of var: "<<&var<<endl;
    cout<<"Address of var: "<<p<<endl;
    cout<<"Address of p: "<<&p<<endl;
    cout<<"Value of var: "<<*p;
    return 0;
}

```

Output:

```

Address of var: 0x7fff5dffc0c
Address of var: 0x7fff5dffc0c
Address of p: 0x7fff5dffc10
Value of var: 101

```

Pointer and arrays

While handling arrays with pointers you need to take care few things. First and very important point to note regarding arrays is that the array name alone represents the base address of array so while assigning the address of array to pointer don't use ampersand sign (&). Do it like this:

Correct: Because arr represent the address of array.

```
p = arr;
```

Incorrect:

```
p = &arr;
```

Example: Traversing the array using Pointers

```

#include <iostream>
using namespace std;
int main(){
    //Pointer declaration
    int *p;
    //Array declaration
    int arr[]={1, 2, 3, 4, 5, 6};
    //Assignment
    p = arr;
    for(int i=0; i<6;i++){
        cout<<*p<<endl;
        //++ moves the pointer to next int position
        p++;
    }
    return 0;
}

```

Output:

```

1
2
3
4
5
6

```

How to increment pointer address and pointer's value?

When we are accessing the value of a variable through pointer, sometimes we just need to increment or decrement the value of variable through it or we may need to move the pointer to next int position (just like we did above while working with arrays). The ++ operator is used for this purpose. One of the examples of ++ operator we have seen above where we traversed the array using pointer by incrementing the pointer value using ++ operator. Let's see few more cases.

```
// Pointer moves to the next int position (as if it was an array)
p++;
// Pointer moves to the next int position (as if it was an array)
++p;

/* All the following three cases are same they increment the value
 * of variable that the pointer p points.
 */
++*p;
++(*p);
++*(p);
```