Pointers

April 28, 2018

1 Definition

Pointers, which are the addresses of variables, can be accessed in C++.

For example in this code snippet:

`int a = 54;`

54 is the value of the variable, in other words, it is the value that is stored in the location reserved the the variable called ’a’.

Now, let’s ask ourselves, where is a? The location of ’a’ can be found using a pointer!

`int a = 54; std::cout<< &a<<"\n"; //This will print the LOCATION of 'a'`

```
In [2]: int a = 54;
        std::cout<<"a = "<<a<<"\n";
        std::cout<<"address of a is at &a = "<< &a<<"\n";
```

```
a = 54
address of a is at &a = 0x7f886e731024
```

But what if we have a pointer and want to access the value stored in that address? That process is called dereferencing, and it is indicated by adding the operator ∗ before the variable’s name. This same operator should be used to declare a variable that is meant to store a pointer.

For example:

```
In [1]: // this is an integer variable with value = 54
        int a = 54;

        // this is a pointer that holds the address of the variable 'a'.
        // if 'a' was a float, rather than int, so should be its pointer.
        int * pointerToA = &a;

        // If we were to print pointerToA, we'd obtain the address of 'a':
        std::cout << "pointerToA stores " << pointerToA << '\n';

        // If we want to know what is stored in this address, we can dereference pointerToA:
        std::cout << "pointerToA points to " << * pointerToA << '\n';
```

```
pointerToA stores 0x7fcf16365028
pointerToA points to 54
```