What is a Pointer?

Get introduced to pointers, the most powerful feature of C++ programming.

We'll cover the following

Introduction#

Suppose your friend asks you for a good source to study data structures from. You will find some good sources and send a hyperlink to them.

Since downloading all the content from the web pages and then sending them in an email requires a lot of memory, you would just send a link to the source. Whenever your friend wants to read the content, they can visit the link and they will be good to go.

Pointers are similar to the hyperlink that stores the location of some other data.

> *In C++, a **pointer** is a variable that stores the address of another variable.*

variable.

Pointer declaration

To declare a variable as a pointer, it's identifier must be preceded by an asterisk `*`. When we use `*` before the identifier, it indicates that the variable being declared is a pointer.

```
DataType  *identifier ;
```

See the code given below!

```
1  #include <iostream>
2
3  using namespace std;
4
5  int main() {
6    // Declares a pointer variable John
7    int *John = nullptr;
8    return 0;
9  }
```

Succeeded

The statement in **Line No.** 7 declares a pointer `John`, and its sole purpose is to store the address of some other variable. Here, `John` only points to the value whose data type is `int`. Therefore, we can say that **John is a pointer to int**

to int.

> 💡 It's considered a best practice to set a pointer to `nullptr` when it is declared, unless it is initialized to some valid address, as we shall see later in this lesson.

> 💡 It's a good practice to use ptr in a pointer's variable name. It indicates that a variable is a pointer, and must be handled differently.

> 📝 If we declare multiple pointers in the same line, we must use an asterisk `*` before each identifier.

Pointer initialization

To initialize a pointer, we must store the address in it. The basic syntax for storing an address of another variable in the pointer variable is given below:

```
ptrVariable = &Variable ;
```

Example program#

Consider the same analogy given in this lesson (https://www.educative.io/collection/page/10370001/6619096843026432/6137072378183680/draft). Let's say `Alice` 's storage house is located at the address **1000**. `John` 's house is located at the address **1004**. `Alice` has stored **5** in its storage house. Whereas, `John` has stored the address of Alice's storage house, which is **1000**.

Here, `John`'s storage house is pointing to `Alice`'s storage house, so `John` is a **pointer**.

Let's translate this example into a C++ program!

```
#include <iostream>

using namespace std;

int main() {
  // Declares a variable Alice
  int Alice = 5;
  // Declares a pointer variable John that can point to int value
  int *John;
  // Stores the address of Alice in John
  John = &Alice;
  // Prints value of Alice
  cout << "Value of Alice = " << Alice << endl;
  // Prints value (address of Alice) of John
  cout << "Value of John = " << John << endl;

  return 0;
}
```

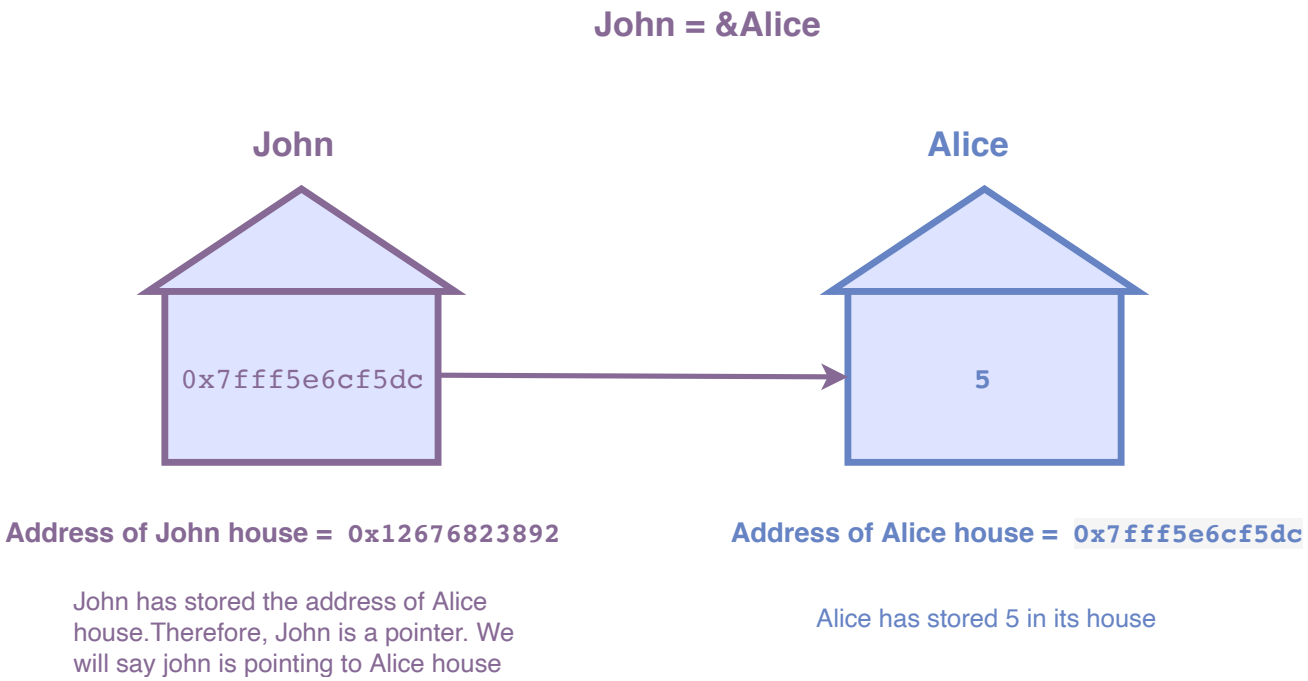
```
Output                                                    0.92s

 Value of Alice = 5
 Value of John = 0x7ffc3d6efa7c
```

Explanation#

Line No. 11: `John` is a pointer, and `&Alice` gives us the memory address of `Alice`. Therefore, we are storing the address of `Alice` in `John`. We can say `John` is pointing to `Alice`.

Using ampersand `&` is like getting the address of Alice instead of seeing what Alice has stored in its storage house.



Pointer

> 📝 An error occurs when the pointer points to the variable of a different data type.

Null pointer#

If the pointer is pointing to nothing, then it should be initialized to **nullptr**. It is known as a null pointer. The value of the null pointer is **0**.

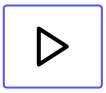
📝 If we don't initialize a pointer, it is automatically initialized to **0**.

Press the **RUN** button and see the output!

```
#include <iostream>

using namespace std;

int main() {
  int *John = nullptr;
  cout << John;
}
```




```
Output                                                    0.89s

 0
```

In the code above, we initialize the `John` to **nullptr**. It means `John` is pointing to nothing.

Quiz

Q Which of the following will declare pointer `valuePtr` and `outputPtr` of

type `int` ?

You can select multiple correct answers.

Selected Option

A)

```
int *valuePtr;
int *outputPtr;
```

B)

```
int *valuePtr , outputPtr;
```

Explanation

If we declare multiple pointers in the same line, we must use an asterisk `*` before each identifier.

Selected Option

C)

```
int *valuePtr , *outputPtr;
```

D) All of the above

Submit Answer

Reset Quiz

So far, we have seen the basic syntax for declaring and initializing a pointer. Now let's see how to access the value pointed by the pointer in C++.