



# Address-of Operator

Get acquainted with the address-of operator in C++.

We'll cover the following

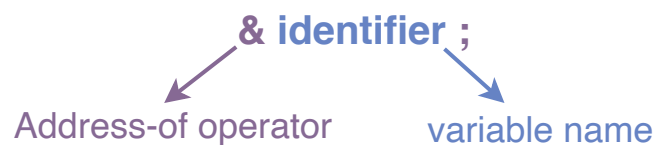


- Address-of operator
  - Example program
  - Explanation

## Address-of operator#

We now know that when we declare a variable, the compiler allocates space someplace in the memory location. What if we want to know the memory address where the variable has been allocated memory?

For this, we will use the address-of operator & before the identifier to access the address of the variable.



---

*The **address-of operator (&)** is a unary operator. It is used to extract the memory address of the variable.*

---



## Example program#

Consider the same analogy given in this lesson

(<https://www.educative.io/collection/page/10370001/6619096843026432/6137072378183680>). Let's say John's storage house is located somewhere in the **Memory Society**, and he has stored **10** in his house.

Let's write a C++ program to find the address of John house.

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

```
1 #include <iostream>
2
3 using namespace std;
4
5 int main() {
6     // Declare a variable John
7     int John = 10;
8     // Prints the memory address in which value of John is stored
9     cout << "John Address = " << &John << endl;
10    // Prints the value of John
11    cout << "John Value = " << John << endl;
12    return 0;
13 }
```



Output


1.55s


```
John Address = 0x7ffde1daafb4
John Value = 10
```

# Explanation#




**Line No. 7:** Stores 10 in John

 **Line No. 9:** Shows us the address of John

 **0x** at the start of the memory address shows that the address is in hexadecimal format.

**Line No. 11:** Displays the value stored in John

 The **memory address** depends upon your machine. Therefore, if you run the same program on a different machine, you will get a different memory address.

We can access the variable address using the address-of (&) operator. Here, we have noticed that memory cells can store numbers, and addresses are numbers, too. So can we declare a variable and store the address of another variable in it?

## Quiz



Q The second statement in the following code snippet does which of the following?

```
int num = 10 , numAddress ;  
numAddress = &num;
```



☐ A) Stores the address of num in numAddress

☐ B) Stores the value of num in numAddress

Your Answer



C) Generates an error

Explanation

We cannot store the address of some variable in the value of another variable.

Submit Answer

Reset Quiz ↻

Let's explore the functionality of pointers in the upcoming lesson.

Stay tuned!

← Back

Variables and Memory

Next →

What is a Pointer?



Mark as Completed



Report an Issue