# The Address Operator

**The address operator, `&`, when placed in front of a variable, returns the address where** the variable is stored in memory.

```
cout << "&a->" << &a << endl;
cout << "&b->" << &b << endl;
```

Addresses are normally printed in hexadecimal, and depend on the size of the pointer. Here's the output from this fragment of code when run on two platforms:

```
&a->009CF808 - Visual C++ 19 (Windows)
&b->009CF7F8
&a->0x7fff448e448c - G++ & Clang (Unix)
&b->0x7fff448e4490
```

Notice that Visual Studio has 32-bit addresses, while Unix uses 64-bit. Of course, the actual values of the addresses printed on your machine will be entirely different. You can take the address of a variable, such as **`&a`** or **`&b`**, **but not a type**. Writing **`&int`** is nonsensical and illegal.

You also cannot take the address of a literal or expression: **`&12`** is meaningless.