C++ Modifier Types

Previous Page Next Page

C++ allows the **char, int,** and **double** data types to have modifiers preceding them. A modifier is used to alter the meaning of the base type so that it more precisely fits the needs of various situations.

The data type modifiers are listed here −

- signed
- unsigned
- long
- short

The modifiers **signed, unsigned, long,** and **short** can be applied to integer base types. In addition, **signed** and **unsigned** can be applied to char, and **long** can be applied to double.

The modifiers **signed** and **unsigned** can also be used as prefix to **long** or **short** modifiers. For example, **unsigned long int**.

C++ allows a shorthand notation for declaring **unsigned, short,** or **long** integers. You can simply use the word **unsigned, short,** or **long,** without **int**. It automatically implies **int**. For example, the following two statements both declare unsigned integer variables.

```
unsigned x;
unsigned int y;
```

To understand the difference between the way signed and unsigned integer modifiers are interpreted by C++, you should run the following short program −

Live Demo

```
#include <iostream>
using namespace std;

/* This program shows the difference between
   * signed and unsigned integers.
*/
int main() {
   short int i;           // a signed short integer
   short unsigned int j;  // an unsigned short integer

   j = 50000;

   i = j;
   cout << i << " " << j;

   return 0;
}
```

When this program is run, following is the output −

```
-15536 50000
```

The above result is because the bit pattern that represents 50,000 as a short unsigned integer is interpreted as -15,536 by a short.

Type Qualifiers in C++

The type qualifiers provide additional information about the variables they precede.

Sr.No	Qualifier & Meaning
1	**const** Objects of type **const** cannot be changed by your program during execution.
2	**volatile** The modifier **volatile** tells the compiler that a variable's value may be changed in ways not explicitly specified by the program.
3	**restrict** A pointer qualified by **restrict** is initially the only means by which the object it points to can be accessed. Only C99 adds a new type qualifier called restrict.

Previous Page Next Page