

Операции инкремента и декремента в C++

Инкремент ++ – это увеличение на единицу.

Декремент -- – это уменьшение на единицу.

Операции декремента и инкремента с лёгкостью заменяются арифметическими операциями или операциями присваивания. Но использовать операции инкремента и декремента намного удобнее.

```
//синтаксис операций инкремента и декремента
++/*имя переменной*/; // префиксный инкремент
1 (преинкремент)
2 /*имя переменной*/++; // постфиксный инкремент
3 (постинкремент)
4 --/*имя переменной*/; // префиксный декремент
5 (предекремент)
/*имя переменной*/--; // постфиксный декремент
++/*имя переменной*/; // постфиксный декремент (постдекремент)
```

/*имя переменной*/++; // постфиксный инкремент (постинкремент)

--/*имя переменной*/; // префиксный декремент (предекремент)

/*имя переменной*/--; // постфиксный декремент (постдекремент)

Если операция инкремента или декремента ставится перед именем переменной, то такая операция **называется префиксным инкрементом** (сокращённо — преинкрементом) или **префиксным декрементом** (сокращённо — предекрементом).

Если операция инкремента или декремента ставится после имени переменной, то такая операция называется операцией **постфиксного инкремента** (сокращённо — постинкремент) или **постфиксного декремента** (сокращённо — постдекремент).

При использовании **операции преинкремента** значение переменной, сначала, увеличивается на 1, а затем используется в выражении.

При использовании **операции постинкремента** значение переменной сначала используется в выражении, а потом увеличивается на 1.

При использовании **операции преддекремента**, значение переменной, сначала, уменьшается на 1, а затем используется в выражении.

При использовании **операции постдекремента**, значение переменной, сначала, используется в выражении, а потом уменьшается на 1.

Операции инкремента и декремента в C++

Операция	Обозначение	Пример	Краткое пояснение
преинкремент	++	<code>cout << ++value;</code>	Значение в переменной <code>value</code> увеличивается, после чего оператор <code>cout</code> печатает это значение
предекремент	--	<code>cout << --value;</code>	Значение в переменной <code>value</code> уменьшается, после чего оператор <code>cout</code> печатает это значение
постинкремент	++	<code>cout << value++;</code>	Оператор <code>cout</code> печатает значение переменной <code>value</code> , затем увеличивает это значение на 1
постдекремент	--	<code>cout << value--;</code>	Оператор <code>cout</code> печатает значение переменной <code>value</code> , затем уменьшает это значение на 1

Разработаем программу на основе выражений из таблицы, которая наглядно покажет, как себя будут вести операции

инкремента и декремента

```
3
4  #include <iostream>
5  using namespace std;
6
7  int main(int argc, char* argv[])
8  {
9  int value = 2020;
10 cout << "value = " << value << endl; // начальное значение
11 cout << "++value = " << ++value << endl; // операция преинкремента
12 cout << "value++ = " << value++ << endl; // операция постинкремента
13 cout << "value = " << value << endl; // конечное значение в переменной value
    после выполнения операции постинкремента
14 cout << "--value = " << --value << endl; // операция преддекремента
15 cout << "value-- = " << value-- << endl; // операция постдекремента
16 cout << "value = " << value << endl; // конечное значение в переменной value
    после выполнения операции постдекремента
17 system("pause");
18 return 0;
19 }
```

Результат работы программы

```
value = 2020  
++value = 2021  
value++ = 2021  
value = 2022  
--value = 2021  
value-- = 2021  
value = 2020
```

Для продолжения нажмите любую клавишу . . . █

Эквивалентные операции

`/*операция*/ ++value; /*эквивалентна операции*/ value += 1;`

`/*операция*/ --value; /*эквивалентна операции*/ value -= 1;`

**`/*операция*/ ++value; /*эквивалентна операции*/ value = value
+1;`**

`/*операция*/ --value; /*эквивалентна операции*/ value = value - 1;`