

C++



Prepared by: Mohamed Ayman



facebook.com/sw.eng.MohamedAyman



sw.eng.MohamedAyman@gmail.com



wuzzuf.net/me/engMohamedAyman



codeforces.com/profile/Mohamed_Ayman



Basic Operators

Outline



- 1) Types of Operator
- 2) C++ Arithmetic Operators
- 3) C++ Relational Operators
- 4) C++ Logical Operators
- 5) C++ Bitwise Operators
- 6) C++ Assignment Operators
- 7) C++ Miscellaneous Operators
- 8) C++ Operators Precedence

Types of Operator



- An operator is a symbol that tells the compiler to perform specific mathematical or logical manipulations. C++ is rich in built-in operators and provide the following types of operators:
 - Arithmetic Operators
 - Relational Operators
 - Logical Operators
 - Bitwise Operators
 - Assignment Operators
 - Miscellaneous Operators

C++ Arithmetic Operators



Operator	Description	Example
+	Adds two operands	A + B will give 30
-	Subtracts second operand from the first	A - B will give -10
*	Multiplies both operands	A * B will give 200
/	Divides numerator by denominator	B / A will give 2
%	Modulus Operator and remainder of after an integer division	B % A will give 0
++	Increment operator, increases integer value by one	A++ will give 11
--	Decrement operator, decreases integer value by one	A-- will give 9

C++ Arithmetic Operators Example



- Source code: <https://repl.it/repls/OrderlyDimgreyBuffalo>

```
1  #include <iostream>
2  using namespace std;
3
4  int main()
5  {
6      int a = 21, b = 10, c;
7
8      c = a + b;
9      cout << "Line 1 - value of c is " << c << '\n';
10
11     c = a - b;
12     cout << "Line 2 - value of c is " << c << '\n';
13
14     c = a * b;
15     cout << "Line 3 - value of c is " << c << '\n';
16
17     c = a / b;
18     cout << "Line 4 - value of c is " << c << '\n';
19
20     c = a % b;
21     cout << "Line 5 - value of c is " << c << '\n';
22
23     float d = float(a) / float(b);
24     cout << "Line 6 - value of d is " << d << '\n';
25 }
```

```
Line 1 - value of c is 31
Line 2 - value of c is 11
Line 3 - value of c is 210
Line 4 - value of c is 2
Line 5 - value of c is 1
Line 6 - value of d is 2.1
```

C++ Relational Operators



Operator	Description	Example
<code>==</code>	Checks if the values of two operands are equal or not, if yes then condition becomes true.	<code>(A == B)</code> is not true.
<code>!=</code>	Checks if the values of two operands are equal or not, if values are not equal then condition becomes true.	<code>(A != B)</code> is true.
<code>></code>	Checks if the value of left operand is greater than the value of right operand, if yes then condition becomes true.	<code>(A > B)</code> is not true.
<code><</code>	Checks if the value of left operand is less than the value of right operand, if yes then condition becomes true.	<code>(A < B)</code> is true.
<code>>=</code>	Checks if the value of left operand is greater than or equal to the value of right operand, if yes then condition becomes true.	<code>(A >= B)</code> is not true.
<code><=</code>	Checks if the value of left operand is less than or equal to the value of right operand, if yes then condition becomes true.	<code>(A <= B)</code> is true.

C++ Relational Operators Example



- Source code: <https://repl.it/repls/AbsoluteScalyMouse>

```
1  #include <iostream>
2  using namespace std;
3
4  int main()
5  {
6      int a = 21, b = 10;
7      bool c;
8
9      c = (a == b);
10     cout << "Line 1 - value of c is " << c << '\n';
11
12     c = (a != b);
13     cout << "Line 2 - value of c is " << c << '\n';
14
15     c = (a > b);
16     cout << "Line 3 - value of c is " << c << '\n';
17
18     c = (a >= b);
19     cout << "Line 4 - value of c is " << c << '\n';
20
21     c = (a < b);
22     cout << "Line 5 - value of c is " << c << '\n';
23
24     c = (a <= b);
25     cout << "Line 6 - value of c is " << c << '\n';
26 }
```

```
Line 1 - value of c is 0
Line 2 - value of c is 1
Line 3 - value of c is 1
Line 4 - value of c is 1
Line 5 - value of c is 0
Line 6 - value of c is 0
```




C++ Logical Operators

Operator	Description	Example
&&	Called Logical AND operator. If both the operands are non-zero, then condition becomes true.	(A && B) is false.
	Called Logical OR Operator. If any of the two operands is non-zero, then condition becomes true.	(A B) is true.
!	Called Logical NOT Operator. Use to reverses the logical state of its operand. If a condition is true, then Logical NOT operator will make false.	!(A && B) is true.

C++ Bitwise Operators



Operator	Description	Example
&	Binary AND Operator copies a bit to the result if it exists in both operands.	(A & B) will give 12 which is 0000 1100
	Binary OR Operator copies a bit if it exists in either operand.	(A B) will give 61 which is 0011 1101
^	Binary XOR Operator copies the bit if it is set in one operand but not both.	(A ^ B) will give 49 which is 0011 0001
~	Binary Ones Complement Operator is unary and has the effect of 'flipping' bits.	(~A) will give -61 which is 1100 0011 in 2's complement form due to a signed binary number.
<<	Binary Left Shift Operator. The left operands value is moved left by the number of bits specified by the right operand.	A << 2 will give 240 which is 1111 0000
>>	Binary Right Shift Operator. The left operands value is moved right by the number of bits specified by the right operand.	A >> 2 will give 15 which is 0000 1111

C++ Bitwise Operators Example



- Source code: <https://repl.it/repls/GiddyMistyFoal>

```
1  #include <iostream>
2  using namespace std;
3
4  int main()
5  {
6      int a = 60, b = 13, c;
7
8      c = (a & b);
9      cout << "Line 1 - value of c is " << c << '\n';
10
11     c = (a | b);
12     cout << "Line 2 - value of c is " << c << '\n';
13
14     c = (a ^ b);
15     cout << "Line 3 - value of c is " << c << '\n';
16
17     c = ~a;
18     cout << "Line 4 - value of c is " << c << '\n';
19
20     c = a << 2;
21     cout << "Line 5 - value of c is " << c << '\n';
22
23     c = a >> 2;
24     cout << "Line 6 - value of c is " << c << '\n';
25 }
```

```
Line 1 - value of c is 12
Line 2 - value of c is 61
Line 3 - value of c is 49
Line 4 - value of c is -61
Line 5 - value of c is 240
Line 6 - value of c is 15
```

C++ Assignment Operators



Operator	Description	Example
=	Simple assignment operator, Assigns values from right side operands to left side operand.	<code>C = A + B</code> will assign value of <code>A + B</code> into <code>C</code>
<code>+=</code>	Add AND assignment operator, It adds right operand to the left operand and assign the result to left operand.	<code>C += A</code> is equivalent to <code>C = C + A</code>
<code>-=</code>	Subtract AND assignment operator, It subtracts right operand from the left operand and assign the result to left operand.	<code>C -= A</code> is equivalent to <code>C = C - A</code>
<code>*=</code>	Multiply AND assignment operator, It multiplies right operand with the left operand and assign the result to left operand.	<code>C *= A</code> is equivalent to <code>C = C * A</code>
<code>/=</code>	Divide AND assignment operator, It divides left operand with the right operand and assign the result to left operand.	<code>C /= A</code> is equivalent to <code>C = C / A</code>
<code>%=</code>	Modulus AND assignment operator, It takes modulus using two operands and assign the result to left operand.	<code>C %= A</code> is equivalent to <code>C = C % A</code>



C++ Assignment Operators

Operator	Description	Example
<code><<=</code>	Left shift AND assignment operator.	<code>C <<= 2</code> is same as <code>C = C << 2</code>
<code>>>=</code>	Right shift AND assignment operator.	<code>C >>= 2</code> is same as <code>C = C >> 2</code>
<code>&=</code>	Bitwise AND assignment operator.	<code>C &= 2</code> is same as <code>C = C & 2</code>
<code>^=</code>	Bitwise exclusive OR and assignment operator.	<code>C ^= 2</code> is same as <code>C = C ^ 2</code>
<code> =</code>	Bitwise inclusive OR and assignment operator.	<code>C = 2</code> is same as <code>C = C 2</code>

C++ Assignment Operators Example



- Source code: <https://repl.it/repls/ProductiveMoistMarten>

```
1  #include <iostream>
2  using namespace std;
3
4  int main()
5  {
6      int a = 21, b = 10, c = 0;
7
8      c += a;
9      cout << "Line 1 - value of c is " << c << '\n';
10     c -= b;
11     cout << "Line 2 - value of c is " << c << '\n';
12     c *= a;
13     cout << "Line 3 - value of c is " << c << '\n';
14     c /= b;
15     cout << "Line 4 - value of c is " << c << '\n';
16     c %= a;
17     cout << "Line 5 - value of c is " << c << '\n';
18     c <<= b;
19     cout << "Line 6 - value of c is " << c << '\n';
20     c |= a;
21     cout << "Line 7 - value of c is " << c << '\n';
22     c >>= b;
23     cout << "Line 8 - value of c is " << c << '\n';
24     c &= a;
25     cout << "Line 9 - value of c is " << c << '\n';
26 }
```

```
Line 1 - value of c is 21
Line 2 - value of c is 11
Line 3 - value of c is 231
Line 4 - value of c is 23
Line 5 - value of c is 2
Line 6 - value of c is 2048
Line 7 - value of c is 2069
Line 8 - value of c is 2
Line 9 - value of c is 0
```

C++ Miscellaneous Operators



Operator	Description
sizeof	sizeof operator returns the size of a variable. For example, sizeof(a), where 'a' is integer, and will return 4.
Condition ? X : Y	Conditional operator (?). If Condition is true then it returns value of X otherwise returns value of Y.
,	Comma operator causes a sequence of operations to be performed. The value of the entire comma expression is the value of the last expression of the comma-separated list.
. (dot) and -> (arrow)	Member operators are used to reference individual members of classes, structures, and unions.
Cast	Casting operators convert one data type to another. For example, int(2.2000) would return 2.
&	Pointer operator '&' returns the address of a variable. For example &a; will give actual address of the variable.
*	Pointer operator * is pointer to a variable. For example *var; will pointer to a variable var.

C++ Operators Precedence



Category	Operator	Associativity
Postfix	() [] -> . ++ --	Left to right
Unary	+ - ! ~ ++ -- (type)* & sizeof	Right to left
Multiplicative	* / %	Left to right
Additive	+ -	Left to right
Shift	<< >>	Left to right
Relational	< <= > >=	Left to right
Equality	== !=	Left to right

C++ Operators Precedence



Category	Operator	Associativity
Bitwise AND	&	Left to right
Bitwise XOR	^	Left to right
Bitwise OR		Left to right
Logical AND	&&	Left to right
Logical OR		Left to right
Conditional	?:	Right to left
Assignment	= += -= *= /= %>>= <<= &= ^= =	Right to left
Comma	,	Left to right

C++ Operators Precedence Example



- Source code: <https://repl.it/repls/MadeupLastQuetzal>

```
1  #include <iostream>
2  using namespace std;
3
4  int main()
5  {
6      int a = 20, b = 10, c = 15, d = 5, e;
7
8      e = (a + b) * c / d;
9      cout << "Line 1 - value of e is " << e << '\n';
10
11     e = ((a + b) * c) / d;
12     cout << "Line 2 - value of e is " << e << '\n';
13
14     e = (a + b) * (c / d);
15     cout << "Line 3 - value of e is " << e << '\n';
16
17     e = a + (b * c) / d;
18     cout << "Line 4 - value of e is " << e << '\n';
19
20     e = a + b >> c / d;
21     cout << "Line 5 - value of e is " << e << '\n';
22
23     e = a + b * c / -d;
24     cout << "Line 6 - value of e is " << e << '\n';
25 }
```

```
Line 1 - value of e is 90
Line 2 - value of e is 90
Line 3 - value of e is 90
Line 4 - value of e is 50
Line 5 - value of e is 3
Line 6 - value of e is -10
```

Practice

- Take as an input x and get the result from this equation
- $F(x) = (x-1)^3 + (x+1)^2 + 2x + 7$;
such that $x \geq 0$
- Test Cases:

0
7

1
13

2
21

3
37

4
67

5
117

10
877

20
7347



Practice Solution



- Source code: <https://repl.it/repls/ScientificImportantAntipodesgreenparakeet>

```
1  #include <iostream>
2  using namespace std;
3
4  int main()
5  {
6      int x;
7      cin >> x;
8      int res = (x-1)*(x-1)*(x-1) + (x+1)*(x+1) + 2*x + 7;
9      cout << res;
10 }
```

Practice



- Take as an input x and get the result from this equation between $[0,10]$
- $F(x) = (x-1)^3 + (x+1)^2 + 2x + 7$;
such that $x \geq 0$
- Test Cases:

0
7

1
2

2
10

3
4

4
1

5
7

10
8

20
10

Practice Solution



- Source code: <https://repl.it/repls/MediumturquoiseMaleLeafbird>

```
1  #include <iostream>
2  using namespace std;
3
4  int main()
5  {
6      int x;
7      cin >> x;
8      int res = (x-1)*(x-1)*(x-1) + (x+1)*(x+1) + 2*x + 7;
9      cout << res % 11;
10 }
```

Practice



- You are given a rectangular board of $M \times N$ squares.
- You are given an unlimited number of standard domino pieces of 2×1 squares.
- You are asked to place as many dominoes as possible on the board so as to meet the following conditions:
 - Each domino completely covers two squares.
 - No two dominoes overlap.
 - Each domino lies entirely inside the board
- Find the maximum number of dominoes, which can be placed under these restrictions such that $N, M > 0$
- Test Cases:

4 4
8

5 5
12

4 6
12

5 7
17

4 7
14

5 6
15

Practice Solution



- Source code: <https://repl.it/repls/HastyDimJenny>

```
1  #include <iostream>
2  using namespace std;
3
4  int main()
5  {
6      int n, m;
7      cin >> n >> m;
8      cout << n * m / 2;
9  }
```


Practice

- Theatre Square has a rectangular shape with the size $n \times m$ meters.
- A decision was taken to pave the Square with square granite flagstones. Each flagstone is of the size $a \times a$.
- What is the least number of flagstones needed to pave the Square?
- It's allowed to cover the surface larger than the Theatre Square, but the Square has to be covered.
- It's not allowed to break the flagstones.
- Input will be : $n \ m \ a$ such that $n, m, a > 0$
- Test Cases:

4 4 2
4

5 5 2
9

4 4 3
4

5 5 3
4

5 6 3
4

5 6 2
9



Practice Solution



- Source code: <https://repl.it/repls/ConsideratePlumAegeancat>

```
1  #include <iostream>
2  using namespace std;
3
4  int main()
5  {
6      int n, m, a;
7      cin >> n >> m >> a;
8      int w = (n + a - 1) / a;
9      int h = (m + a - 1) / a;
10     cout << w * h;
11 }
```

Practice



- For a positive integer n let's define a function f :
- $f(n) = -1 + 2 - 3 + \dots + (-1)^n n$
- Your task is to calculate $f(n)$ for a given n , such that $n > 0$ and integer
- Test Cases:

1 -1	6 3
2 1	7 -4
3 -2	8 4
4 2	9 -5
5 -3	10 5

Practice Solution



- Source code: <https://repl.it/repls/HarmlessEnviousWoodcock>

```
1  #include <iostream>
2  using namespace std;
3
4  int main()
5  {
6      int n;
7      cin >> n;
8      cout << n/2 - n*(n%2);
9  }
```



Questions ?

References



Online Courses YouTube playlists:

C++ Documentation

CPP For School

C++ Language Tutorial

C++ Language Tutorial

C++ Tutorial Point

Fundamentals of C++ Programming

Teach Yourself C++ in 21 Days

A Complete Guide to Programming in C++

<http://bit.ly/2kAPL5K>

<http://bit.ly/1flmcHO>

<http://bit.ly/2kifMdj>

<http://bit.ly/1kyBMdz>

<http://bit.ly/2rzE4hQ>

<http://bit.ly/2BGFeO0>

<http://bit.ly/2rJHhyl>

<http://bit.ly/1JXhDtL>

<http://bit.ly/2dVkGY9>