

C++

پریسا حامد روح بخش

موسسه ی پارس پژوهان

فصل سوم

- Operators
- Operator Precedence

Operators

- Operators **are used to perform** operations **on** variables **and** values.
- Assignment operator (=)
- Arithmetic operators (+, -, *, /, %)
- Compound assignment (+=, -=, *=, /=, %=, >>=, <<=, &=, ^=, |=)
- Increment and decrement (++, --)
- Relational and comparison operators (==, !=, >, <, >=, <=)
- Logical operators (!, &&, ||)

Assignment operator

- Assignment operators are used to assign values to variables.

```
// assignment operator
#include <iostream>
using namespace std;

int main ()
{
    int a, b;           // a:?, b:?
    a = 10;             // a:10, b:?
    b = 4;              // a:10, b:4
    a = b;              // a:4, b:4
    b = 7;              // a:4, b:7

    cout << "a:";
    cout << a;
    cout << " b:";
    cout << b;
}
```

Arithmetic operators

$x = 11 \% 3;$

$y = x + 2;$

$z = 10 - 2 ;$

operator	description
+	addition
-	subtraction
*	multiplication
/	division
%	modulo

Compound assignment

expression	equivalent to...
<code>y += x;</code>	<code>y = y + x;</code>
<code>x -= 5;</code>	<code>x = x - 5;</code>
<code>x /= y;</code>	<code>x = x / y;</code>
<code>price *= units + 1;</code>	<code>price = price * (units+1);</code>

```
// compound assignment operators
#include <iostream>
using namespace std;

int main ()
{
    int a, b=3;
    a = b;
    a+=2;           // equivalent to a=a+2
    cout << a;
}
```

Increment and decrement

Example 1	Example 2
<pre>x = 3; y = ++x; // x contains 4, y contains 4</pre>	<pre>x = 3; y = x++; // x contains 4, y contains 3</pre>

Relational and comparison operators

operator	description
==	Equal to
!=	Not equal to
<	Less than
>	Greater than
<=	Less than or equal to
>=	Greater than or equal to

Two **expressions** can be **compared** using relational and equality operators.



```
(7 == 5) // evaluates to false
(5 > 4) // evaluates to true
(3 != 2) // evaluates to true
(6 >= 6) // evaluates to true
(5 < 5) // evaluates to false
```


Logical operators

```
!(5 == 5) // evaluates to false because the expression at its right (5 == 5) is true
!(6 <= 4) // evaluates to true because (6 <= 4) would be false
!true     // evaluates to false
!false    // evaluates to true
```

OPERATOR (or)		
a	b	a b
true	true	true
true	false	true
false	true	true
false	false	false

&& OPERATOR (and)		
a	b	a && b
true	true	true
true	false	false
false	true	false
false	false	false

Operator Precedence

Precedence	Operator	Description	Associativity
1	<code>::</code>	Scope resolution	Left-to-right →
2	<code>a++ a--</code> <code>type() type{}</code> <code>a()</code> <code>a[]</code> <code>. -></code>	Suffix/postfix increment and decrement Functional cast Function call Subscript Member access	
3	<code>++a --a</code> <code>+a -a</code> <code>! ~</code> <code>(type)</code> <code>*a</code> <code>&a</code> <code>sizeof</code> <code>co_await</code> <code>new new[]</code> <code>delete delete[]</code>	Prefix increment and decrement Unary plus and minus Logical NOT and bitwise NOT C-style cast Indirection (dereference) Address-of Size-of ^[note 1] await-expression (C++20) Dynamic memory allocation Dynamic memory deallocation	Right-to-left ←

Operator Precedence

4	. * ->*	Pointer-to-member	Left-to-right →
5	a*b a/b a%b	Multiplication, division, and remainder	
6	a+b a-b	Addition and subtraction	
7	<< >>	Bitwise left shift and right shift	
8	<=>	Three-way comparison operator (since C++20)	
9	< <= > >=	For relational operators < and ≤ and > and ≥ respectively	
10	== !=	For equality operators = and ≠ respectively	
11	a&b	Bitwise AND	
12	^	Bitwise XOR (exclusive or)	
13		Bitwise OR (inclusive or)	
14	&&	Logical AND	
15		Logical OR	

Operator Precedence

16	a?b:c throw co_yield = += -= *= /= %= <<= >>= &= ^= =	Ternary conditional ^[note 2] throw operator yield-expression (C++20) Direct assignment (provided by default for C++ classes) Compound assignment by sum and difference Compound assignment by product, quotient, and remainder Compound assignment by bitwise left shift and right shift Compound assignment by bitwise AND, XOR, and OR	Right-to-left ←
	17	,	Left-to-right →

Common operators

Common operators						
assignment	increment decrement	arithmetic	logical	comparison	member access	other
<pre> a = b a += b a -= b a *= b a /= b a %= b a &= b a = b a ^= b a <<= b </pre>	<pre> ++a --a a++ a-- </pre>	<pre> +a -a a + b a - b a * b a / b a % b ~a a & b a b a ^ b a << b a >> b </pre>	<pre> !a a && b a b </pre>	<pre> a == b a != b a < b a > b a <= b a >= b a <=> b </pre>	<pre> a[b] *a &a a->b a.b a->*b a.*b </pre>	<pre> a(...) a, b a ? b : c </pre>

تمرین

(1) برنامه ای بنویسید که شعاع دایره ای را از کاربر دریافت کند و محیط و مساحت آن را محاسبه و چاپ نماید.

(2) برنامه ای بنویسید که یک عدد صحیح را از ورودی بخواند و مشخص کند زوج است یا فرد.

(3) برنامه ای بنویسید که پنج مقدار صحیح را از ورودی خوانده سپس بزرگ ترین و کوچک ترین مقدار را پیدا کرده و چاپ کند.

(4) برنامه ای بنویسید که سه عدد را از ورودی خوانده و میانگین آن را محاسبه

کند و سپس به کاربر نمایش دهد.



Question ?