principles of programming

Operators

4

Example

What is the output of the following program

```
int main()
{
      int a = 5;
      int b = 3;
      int c = 2;
      float Y = Y = -(b + c) / a;
      cout << "Y is " <<Y << endl;
      return 0;
}
```

C++ Operators

- Assignment operator
  - (=)
- Arithmetic operators
  - (+, -, *, /, %)
- Compound assignment
  - (+=, -=, *=, /=, %=, >>=, <<=, &=, ^=, |=)
- Increment and decrement
  - (++, --)
- Relational and comparison operators
  - (==, !=, >, <, >=, <=)
- Logical operators
  - (!, &&, ||)
- Conditional ternary operator
  - (?)
- Comma operator
  - (,)

Assignment operator (=)

- The assignment operator assigns a value to a variable.

```
// 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(+, -, *, /, %)

- The five arithmetical operations supported by C++ are

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

Compound assignment

(+=, -=, *=, /=, %=, >>=, <<=, &=, ^=, |=)

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

Example

```
// 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

- The decrement operator (--) decrements the value of its operand by 1.
- The increment operator (++) increments the value of its operand by 1.

The prefix version (++x or --x)

- Comes before the operand, as in ++x
- First increments or decrements the variable by 1 and then uses the value of the variable.

```
int x = 5;
int y = ++x;
```

means

1. Change x
2. Then assign to y
3. x=6 y=6

```
int x = 5;
x = x + 1;
int y = x;
```

The postfix version (x++ or x--)

- Comes after the operand, as in x++
- Uses the current value of the variable and then increment or decrements the variable by 1.

```
int z = 5;
int y = z++;
```

means

Assign z to y.
Then change z.
y is 5, z is 6

```
int z = 5;
int y = z;
z = z + 1;
```

Relational and comparison operators

- The result of such an operation is either true or false (i.e., a Boolean value)

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

Example

```
(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
```

```
(a == 5)     // evaluates to false, since a is not equal to 5
(a*b >= c)   // evaluates to true, since (2*3 >= 6) is true
(b+4 > a*c)  // evaluates to false, since (3+4 > 2*6) is false
((b=2) == a) // evaluates to true
```

Logical Operators

- To combine or modify existing expressions.

!	NOT
&&	AND
||	OR

- **Example**

```
a > 5 && b > 5
ch== 'y' || ch== 'Y'
!valid
!(x > 5)
```