University of Engineering and Technology,
Peshawar, Pakistan

CSE 102: Computer Programming

Lecture 2

Expression, Operands, and Operators

By;
Dr. Muhammad Athar Javed Sethi

cout (See Out)

- Console Output, console represents computer display screen.
- Part of iostream header file.
- Quotes around string
- Ends in semicolon
- Syntax of cout is;
  - std::cout << const1/var1 << const2/var2;

cin (See In)

- Console input, console represents computer display screen.
- Part of iostream header file.
- When an input statement is executed the computer waits to receive an input from keyboard. When value is typed and enter key is pressed value is assigned to variable and control shifts to the next line.
- Ends in semicolon
- Syntax of cin is;
  - std::cin >> var1;

Comment statement

- Non executable statement.
- Used to add remarks or comments.
- Compiler ignores the comments given in the program.
- In c++ double slash "//" are used to mark comment statement.
- Another notation can also be used in c++; comments or remarks are enclosed within symbols '/*' and '*/'

Escape sequence (control characters)

- \n – new line (It moves the control to the beginning of the next line)
  - endl manipulator has similar effect
- \t – horizontal tab
- \r – carriage return (it moves the cursor to the beginning of the current line)
- \a – sound bell
- \b– backspace

Additional controls

- \\ – print backslash
- \" – print double quote
- \’ – print single quote

C++ Techniques

- Variables
- Operators
- Decisions
- Loops

Variables

- A variable is a storage location (identified by a memory address) paired with an associated symbolic name (an *identifier*), which contains some quantity of information referred to as a value.
- Name and type must be declared before use.
- Explicit declaration.

Variables

- Variables, can be declared anywhere in the program.
- However they are **only** visible to the program within the block of code in which they are defined.

```
int main() {
    int x = 4;
    if (x>3)
    {
          int y = 3;
    }
    return (0)
}
```

Values and Variables

- Basic Types:
  - Integers (signed and unsigned)
    - short
    - int
  - Floating point numbers
    - float
    - double
  - Characters (char)
  - Boolean (bool)

Basic Types: `int` and `float`

- Integers (`int`)

```
0    1    1000    -1    -10    666
```

- Floating point numbers (`float`)

```
1.0    .1    1.0e-1    1e1
```

Basic Types: `char`

- Characters (`char`)

```
'a' 'z' 'A' 'Z' '?' '@' '0' '9'
```

Naming variables

- Rules For Variables;
  - First character of variable name must be an alphabetic character.
  - Underscore can be used as first character of variable name.
  - Blank space are not allowed in a variable name.
- A good name for your variables is important.

```
int a,b;
double d;
/* These are
not good names
*/
```

```
int start_time;
int no_students;
double course_mark;
/* This is a bit better */
```

- Ideally, a comment with each variable name helps people know what they do.
- case sensitive: x2 & X2 are different

C++ Variable Covered Range

Data Type	Size	Data storage range
`int`	16 bits (2 byte)	-32768 to 32767
`short int`	16 bits (2 bytes)	-32768 to 32767
`long int`	32 bits (4 bytes)	-2147483648 to 2147483647
`unsigned int`	16 bits (2 bytes)	0 to 65535
`unsigned long int`	32 bits (4 bytes)	0 to 4294967295
`float`	32 bits (4 bytes)	$3.4x10^{(-38)}$ to $3.4x10^{(+38)}$
`long float`	64 bits (8 bytes)	$1.7x10^{(-308)}$ to $1.7x10^{(+308)}$
`double`	64 bits (8 bytes)	$1.7x10^{(-308)}$ to $1.7x10^{(+308)}$
`long double`	80 bits (10 bytes)	$3.4x10^{(-4932)}$ to $1.1x10^{(+4932)}$
`char`	8 bits (1 byte)	In case of string storage capacity is 1 byte to 65535 bytes

Expressions

- A constant or variable by itself is an expression
- Combinations of constants and variables with operators are also expressions
- Examples

```
7              /* Constant */
x              /* Variable */
x + 7          /* with operator */
x = x + 7 /* with operators */
x = x + 7;  /* Simple statement */
```

Operators

- + = add
- – = subtract
- * = multiply
- / = division
- % = modulus – remainder

Logical Operators

- C++ defines these logical operators: <, >, <=, >= and == (the equivalence operator)
- You can compare any variable. Characters are compared based on their ASCII values.
- All answers will be true (not zero) or false (0)
- You can extend the logic with && (and), ! (not) and || (or).

More operators

- == test equality x == y
- = assignment x = y
- != not equal
- >= greater than or equal
- <= less than or equal

More operators

- > greater than
- < less than
- no space between operators
- = = is invalid == OK
- & bitwise AND
- | bitwise OR
- ~ bitwise NOT

Compound statements

- && and
- || or
- ! not

The If Statement

- Syntax: if (expression) statement;
- If the expression is true (not zero), the statement is executed. If the expression is false, it is not executed.
- You can group multiple expressions together with braces:

```
if (expression) {
  statement 1;
  statement 2;
  statement 3;
}
```

The If/Else Statement

- Syntax: if (expression) statement_1; else statement_2;
- If the expression is true, statement_1 will be executed, otherwise, statement_2 will be.

```
if (myVal < 3)
   std::cout << "myVal is less than 3.\n";
else
   std::cout << "myVal is greater than or
   equal to 3.\n";
```

Operators and Expressions

- Addition
- Subtraction
- Multiplication
- Real number division
- Integer division
- Logical AND
- Logical OR
- Logical NOT

```
x = y + z;
x = y - z;
x = y * z;
x = y / 3.14;
x = y / 10;
if (x==1 && y==2)
if (x==1 || y==2)
if (!x)
```

C++ Operators

- More Identical operators:

Equal to	`if (x==10)`
Not equal to	`if (x!=10)`
Less than	`if (x<10)`
Greater than	`if (x>10)`
Less than / equal to	`if (x<=10)`
Greater than / equal to	`if (x>=10)`

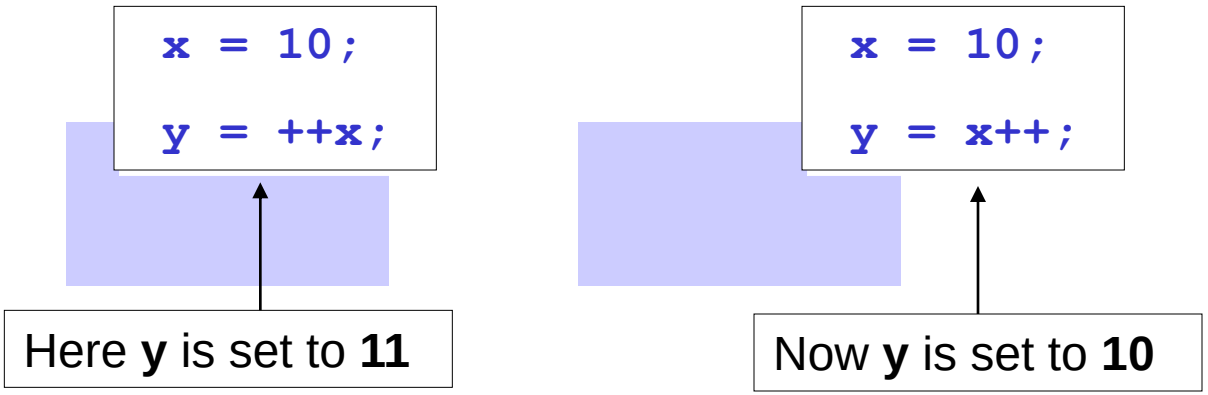
C++ Operators

- The increment Operator
  - Increment `x++ or ++x`
  - Decrement `x-- or --x`
  - Compound Assignment Expression

x*=3 (multiply x by 3)
x+=5 (add 5 to x)
x-=10 (subtract 10 from x)
x/=2 (halve x)

C++ Operators

- There is a subtle difference between x++ and ++x.
- ++x will increment the variable before it does anything else with it, x++ will increment after any assignments.



- In the first two cases x is set to 11, but in the first this is done **before** the assignment.

goto Statement

- Unconditional control transfer statement.
- Used to transfer the control to a specified label in the same program without evaluating any condition
- Syntax of the “goto” statement is;
  - goto label;
  - Label can be any name followed by colon in the program.
  - Any word that can be used as a variable name can be used as a label in c++.

If else

```
if(income > 17000)
    std::cout << “pay tax”;
else
    std::cout << “find a better
  job”;
```

one of these statements always execute

Multiple statements

```
if(x > y)
   {
      x = y + 4;
      z = 2 + y;
    }
```

Conditional operator

```
grade > 90 ? Std::cout <<
  “passed”: std::cout“failed”;
```

- condition to test >90
- after question mark
- 1st statement if condition is true
- 2nd statement if condition is false

Equivalent structure

```
if(grade > 90)
    std::cout << “passed”;
else
    std::cout << “failed”;
```

Another conditional example

```
rate = (hours > 40) ? 3.50 : 0.25;
```

- work > 40 hours
- rate = $3.50 hour
- else $0.25 hour