南方科技大学
SOUTHERN UNIVERSITY OF SCIENCE AND TECHNOLOGY

C/C++ Program Design

CS205

Week 3

Prof. Shiqi Yu (于仕琪)
<yusq@sustech.edu.cn>
Prof. Feng Zheng(郑锋)
<zhengf@sustech.edu.cn>

Some operators

The slides are based on the book <Stephen Prata, C++ Primer Plus, 6th Edition, Addison-Wesley Professional, 2011>

Increment/Decrement Operators

- Prefixing versus postfixing: `++x, x++, --x, x--`
  - Prefix form is more efficient
- The increment/decrement operators and pointers
  - Adding an increment operator to a pointer increases its value by the number of bytes in the type it points to
  - The prefix increment, prefix decrement, and dereferencing operators have the same precedence (from right to left)
  - Postfix increment and decrement operators have the same precedence, which is higher than the prefix precedence(from left to right)
- See program example plus_one.cpp
  - The increment (++) and decrement (--) operators

Combination assignment operators

- Combination assignment operators

Operator	Effect (L=left operand, R=right operand)
+=	Assigns L + R to L
-=	Assigns L - R to L
*=	Assigns L * R to L
/=	Assigns L / R to L
%=	Assigns L % R to L

```
int i = 10;
i += 5;

cout << "i = " << i << endl;
```

- See program example combination.cpp

The ?: Operator

- Conditional operator (question mark)

```
int c;
if (a > b)
    c = a;
else
    c = b;
```

```
int c = a > b ? a : b;
```

- See program example conditional.cpp

`for` loop

The slides are based on the book <Stephen Prata, C++ Primer Plus, 6th Edition, Addison-Wesley Professional, 2011>

for Loops

- Why needs loop operations?
  - Perform repetitive tasks
  - Most tasks have the same process

- See program example forloop.cpp
  - Increment operator: ++ operator (i = i + 1;)

Introducing **for** Loops

```
for (initialization; test-expression; update-expression)
        body;
```

- Parts of a for Loop
  - Setting a value initially
  - Testing whether the loop should continue
  - Executing the loop actions - body
  - Updating value(s) used for the test

Introducing **for** Loops

- See program example num_test.cpp
  - Decrement operator: -- operator (i = i - 1;)

- The range-based for loop (C++11)
  - See Program example cxx11loop.cpp
    - ✓ Colon symbol :
    - ✓ & symbol: reference variable
    - ✓ To modify the array contents

More Examples

- See program example formore.cpp
  - Factorial definition
    - ✓ Zero factorial, written as 0!, is defined to be 1 (exclamation marks!)
    - ✓ The factorial of each integer being the product of that integer with the preceding factorial
- See program example bigstep.cpp
  - Changing the step size

Example: Nested Loops(嵌套) and Two-Dimensional Arrays

The maxtemps array viewed as a table:

		0	1	2	3	4
maxtemps[0]	0	maxtemps[0][0]	maxtemps[0][1]	maxtemps[0][2]	maxtemps[0][3]	maxtemps[0][4]
maxtemps[1]	1	maxtemps[1][0]	maxtemps[1][1]	maxtemps[1][2]	maxtemps[1][3]	maxtemps[1][4]
maxtemps[2]	2	maxtemps[2][0]	maxtemps[2][1]	maxtemps[2][2]	maxtemps[2][3]	maxtemps[2][4]
maxtemps[3]	3	maxtemps[3][0]	maxtemps[3][1]	maxtemps[3][2]	maxtemps[3][3]	maxtemps[3][4]

- Example:

int maxtemps[4][5];

- See program example nested.cpp

Relational Expressions

The slides are based on the book <Stephen Prata, C++ Primer Plus, 6th Edition, Addison-Wesley Professional, 2011>

Expressions

- A C++ expression is a value or a combination of values and operators
- Every C++ expression has a value
  - A for control section uses three expressions
  - Relational expressions such as x < y evaluate to the bool values
  - Evaluating the expression is the primary effect
    - ✓ Evaluating x + 15 calculates a new value, but it doesn't change the value of x
    - ✓ But evaluating ++x + 15 does have a side effect because it involves incrementing x

Relational Expressions

- C++ provides six relational operators to compare numbers
  - Exclamation mark

Operator	Meaning
<	Is less than
<=	Is less than or equal to
==	Is equal to
>	Is greater than
>=	Is greater than or equal to
!=	Is not equal to

Comparisons in Test Expression

- Program example equal.cpp
  - A mistake you'll probably make
  - = or ==
- Program example compstr1.cpp
  - Comparing C-style strings
  - strcmp(str1,str2)
- Program example compstr2.cpp
  - Comparing string class strings
  - Using relational symbol (!=)

`while` loop

The slides are based on the book <Stephen Prata, C++ Primer Plus, 6th Edition, Addison-Wesley Professional, 2011>

The while Loop

- while is entry-condition loop
- It has just a test condition and a body
  - Do something to affect the test-condition expression
- See Program example while.cpp
  - Two types of condition expression

```
while (name[i] != '\0')
while (name[i])
```

for Versus while

- In C++ the for and while loops are essentially equivalent

```
for (init-expression; test-expression; update-expression)
{
    statement(s)
}
```

```
init-expression;
while (test-expression)
{
    statement(s)
    update-expression;
}
```

```
while (test-expression)
    body
```

```
for ( ;test-expression;)
    body
```

More Loops

- The do while Loop
  - It's an exit-condition loop
  - Such a loop always executes at least once
  - See Program example dowhile.cpp

Example: Loops and Text Input

- Using unadorned cin for input
  - When to stop?
    - ✓ A sentinel character
  - See program example textin1.cpp
    - ✓ The program omit the spaces
    - ✓ Program and operating system both work
- cin.get(char) to the rescue
  - See program example textin2.cpp
    - ✓ Read the space
    - ✓ Declare the argument as a reference

Branching Statements

The slides are based on the book <Stephen Prata, C++ Primer Plus, 6th Edition, Addison-Wesley Professional, 2011>

The **if** Statement

- One of the keys to designing intelligent programs is to give them the ability to make decisions
  - Looping
  - if statement

- See program example if.cpp

More than one selections

- The if else Statement
  - Decide which of two statements or blocks is executed
  - Must use braces to collect statements into a single block
  - Remember the conditional compilation #if, #else
- The if else if else Construction
- See program example ifelseif.cpp

Logical Expressions

The slides are based on the book <Stephen Prata, C++ Primer Plus, 6th Edition, Addison-Wesley Professional, 2011>

The Logical OR Operator: ||

- Three operators
  - Logical OR, written ||
  - Logical AND, written &&
  - Logical NOT, written !

The Value of `expr1 || expr2`		
	`expr1 == true`	`expr1 == false`
`expr2 == true`	true	true
`expr2 == false`	true	false

- The logical OR operator: ||
  - || has a lower precedence than the relational operators
  - The || operator is a sequence point
  - C++ won't bother evaluating the expression on the right if the expression on the left is true
  - See program example or.cpp

```
int a = 1, b = 1;
if (a||b++)
{
}
```

AND Operator: && NOT Operator: !

- AND Operator
  - Lower precedence than the relational operators
  - Acts as a sequence point
  - C++ doesn't bother evaluating the right side in some cases
- NOT Operator
  - Exclamation point
  - If expression is true, or nonzero, then !expression is false
  - If expression is false, then !expression is true

The Value of `expr1 && expr2`

	`expr1 == true`	`expr1 == false`
`expr2 == true`	true	false
`expr2 == false`	false	false

Logical Operator Facts

- Precedence
  - The NOT(!) operator has a higher precedence than any of the relational or arithmetic operators
  - The AND operator has a higher precedence than the OR operator
  - Use parentheses to tell the program the interpretation you want

NOT-----relational-----AND-----OR

- The cctype library of character functions
  - A handy package of character-related functions
  - `isalnum() isdigit() isspace()…`

switch Statement

The slides are based on the book <Stephen Prata, C++ Primer Plus, 6th Edition, Addison-Wesley Professional, 2011>

The switch Statement

- Acts as a routing device that tells the computer which line of code to execute next
- You must use the **break**

```
switch (integer-expression)
{
      case label1 : statement(s)
      case label2 : statement(s)
     ...
      default     : statement(s)
}
```

- See program example switch.cpp

More About switch

- switch and if else
  - Let a program select from a list of alternatives
  - A switch statement isn't designed to handle ranges
  - Each switch case label must be a single value
  - Also that value must be an integer
  - A switch statement can't handle floating-point tests

`break` and `continue` Statements

The slides are based on the book <Stephen Prata, C++ Primer Plus, 6th Edition, Addison-Wesley Professional, 2011>

The break and continue Statements

- The break and continue statements enable a program to skip over parts of the code
  - break causes program execution to pass to the next statement following the switch or the loop
  - continue statement is used in loops and causes a program to skip the rest of the body of the loop and then start a new loop cycle
- See program example jump.cpp

Example: Number-Reading Loops

- What happens if the user responds by entering a word instead of a number?

```
int n;
cin >> n;
```

- See program example cinfish.cpp
  - The preceding example doesn't attempt to read any input after non-numeric input

File input & output

The slides are based on the book <Stephen Prata, C++ Primer Plus, 6th Edition, Addison-Wesley Professional, 2011>

Simple File Output

- Main steps for using file output
  - Include the fstream header file
  - Create an ofstream object
  - Associate the ofstream object with a file (C-style) using open()
  - Use the ofstream object in the same manner you would use cout
  - Use the close() method to close the file
- See program example outfile.cpp

Simple File Input

- Main steps for using file input
  - Include the fstream header file and account for the std
  - Declare one or more ifstream variables, or objects
  - Associate a ifstream object with a file using open()
  - Use the close() method to close the file
  - Use >> operator, get(), getline(), …… method
- See program example sumafile.cpp
  - What happens if you attempt to open a non-existent file for input?
  - exit(EXIT_FAILURE);
  - Communicate with the operating system
  - Terminate the program