ILLINOIS INSTITUTE OF TECHNOLOGY

CS 105 Lecture 4

Selection Statements

Wed, Jan 26, 2011, 6:05 pm

Announcements

- Quiz grades to be posted on Blackboard
- Quizzes handed back next Wednesday.

Selection (if-then-else)

- Programming has 3 types of control:
  - Sequential (normal): Control of execution proceeds one after the other.
  - Selection (if-then-else): Control proceeds dependent on conditions.
  - Iteration (looping): Control repeated until condition met.

C++ if Statement

- Syntax: `if (`*`condition`*`)` *`statement`*`;`
- If condition is true, statement is executed
- If condition is false, statement is not executed.
- Statement can be a single statement or a compound statement.

Compound Statement

- To make the statement to do multiple actions, a compound statement is needed

```
if (condition)
{
    statement;
    statement;
}
```

Simple Conditions

- How do we build `if` conditions?
- Simple conditions are built using
  - Relational Operators: < , > , >= , <=
  - Equality Operators: == , !=
- Later we'll see how to build complex conditions.

Selection Examples

```
if (age < 30)
   cout << “You are very very Young” << endl;
if (grade == ‘A’)
   cout << “Congratulations!” << endl;
if (grade != ‘F’)
   cout << “You passed!” << endl;
```

if-else Statement

- The `if-else` statement is a 2-way decision.
- Syntax:

```
if (condition)
{
   statement(s);  //condition true
}
else
{
   statement(s);  //condition false
}
```

if-else Example

```
if (yourGrade >= 60)
{
  cout << "You passed!!" << endl;
}
else
{
  cout << "Uh...." << endl;
}
```

Compound Statements

- Compound statement: Sequence of statements surrounded by curly braces.
- Must use compound statement if more than one statement is supposed to be under control of `if` or `else` conditional.
- Include curly braces after if so if other statements are added in future, curly braces are already there.

Try This: Take 5 min

- Write a program that prompts the user to enter an integer.
    - Store the integer in a variable
    - If the integer entered is greater than 10, display the words “Greater than 10!” to the display
    - If the integer entered is not greater than 10, display the words “Not greater than 10!” to the display

```
/* This program prompts for an integer and prints out
whether it is greater than GREAT_NUM or not */

#include <iostream>
using namespace std;
const int GREAT_NUM = 10;
int
main()
{
   int numEntered;
   cout << "Enter an integer: ";
   cin >> numEntered;
   if (numEntered > GREAT_NUM)
      cout << "Greater than " << GREAT_NUM << "!!";
   else
      cout << "Not greater than " << GREAT_NUM << "!!";
}
```

The Boolean Type

- The two Boolean values are `true` and `false`.
- In C++, Type `bool`. Example:

```
bool done = false;
   . . .
if (currentLetter == 'Z')
   done = true;
   . . .
```

Boolean Expressions

- `if`/`else` conditionals must evaluate to `true` or `false`, and are therefore called <u>boolean expressions</u>.
- In C++, any non-zero value is considered true; any expression evaluating to zero is considered false.
- But usually we use `true` and `false`

Logical Operators

- A Logical Operator is one that manipulates logical values
- && Is Logical “AND”: Both operands must be true for entire expression to be true.
- || Is Logical ‘OR”: One (or both) of the operands must be true for entire expression to be true.

Operator Precedence

- ()
- -, ! (not) (Unary [1-operand] operators)
- *, /, % (Multiplicative operators)
- +, - (Addition operators)
- <, <=, >, >= (Relational operators)
- ==, != (Equality operators)
- && (Logical AND)
- || (Logical OR)
- = (Assignment)