Making Decisions

* With only the sequence structure, all instructions are executed in order
* A selection structure is needed when a decision must be made (based on some condition) before selecting an instruction to execute
* A selection structure’s condition must be phrased as a Boolean expression (evaluates to true or false)
* Single-alternative selection structure
  + Set of instructions is executed only if condition evaluates to true
* Dual-alternative selection structure
  + Executes different sets of instructions based on whether condition evaluates to true or false
* True path
  + Instructions followed when condition evaluates to true
* False path
  + Instructions followed when condition evaluates to false

Coding a Selection Structure in C++

* The if (and else) statement is used to code most selection structures in C++
* Syntax

if (condition)

one or more statements (true path)

[else

one or more statements (false path)]

* Keyword if and *condition* are required
* Portion in brackets (else cause) is optional
  + Only used for dual-alternative selection structures

Comparison Operators

* Comparison operators are used to compare two values that have the same data type
  + Less than (<)
  + Less than or equal to (<=)
  + Greater than (>)
  + Greater than or equal to (>=)
  + Equal to (==)
  + Not equal to (!=)
* No spaces between dual-character symbols

Logical Operators

* **Logical Operators** allow you to combine two or more conditions (sub-conditions) into one compound condition
* Also called as Boolean operators (always evaluate to true or false)
* Two most common are And (&&) and Or (||)
* All sub-conditions must be true for a compound condition using And to be true
* Only one of the sub-conditions must be true for a compound using Or to be true

Getting Data from the Keyboard (cont’d.)

HOW TO Use cin and >> to get numeric or character data

**Syntax:**

cin >> variableName;

variableName = constant (right side set to left side; right to left)

**Example**

double price = 0.0;

cin >> price;

Arithmetic Operations in C++

* You can evaluate arithmetic expressions in C++ using arithmetic operators
* Operators are negation (-), addition (+), subtraction (-), multiplication (\*), division (/), and modulus (%)
* Negation is a unary operator (one operand) and subtraction is a binary operator (two operands)
* Modulus supplies the remainder
  + 8 % 2 = 0
  + 7 % 2 = 1
  + 8 % 3 = 2
* Arithmetic Assignments are a faster way to say variableName (arithmetic operation) constant
* Rate += .5 is the same as Rate = Rate + .5

Static\_Cast

* Static\_cast
* Used to explicitly convert data from one data type to another
* Called an **explicit type conversion** or **type cast**
* Syntax: static\_cast<*datatype>* (data)
  + Data can be a literal constant, named constant, or variable
  + dataType is the data type to which you want the *data* converted

C++ Terms/Operations

* cin >> Used to have the user input something into a variable through the use of the keyboard
  + cin >> variableName >>endl;
* cout << Used to display something onto the screen for the user to read
  + cout << “Hello user, I am the programmer” << endl;
* endl; Used to end the line of text
* // Used to include a comment that is not ran by the program; for programmer eyes
* #include <iostream> Used to tell the program to include commands such as cin and cout