Computer Science I

Chapter 4 - Making Decisions

Hanoi University of Science and Technology

November 5, 2025

Slide contents are mainly based on the following documents:

- Tony Gaddis, Starting out with C++, Six Edition, Pearson, 2009
- Tony Gaddis, Power Points Slides:
  `https://www.umsl.edu/~antognolij/gaddis.html`

Logical Operators

- Used to create relational expressions from other relational expressions

Operator	Meaning	Description
&&	AND	New relational expression is true if both expressions are true
||	OR	New relational expression is true if either expression is true
!	NOT	Reverses the value of an expression – true expression becomes false, and false becomes true

Logical Operators

- `int x = 12, y = 5, z = -4;`

`(x > y) && (y > z)`	`true`
`(x > y) && (z > y)`	`false`
`(x <= z) || (y == z)`	`false`
`(x <= z) || (y != z)`	`true`
`!(x >= z)`	`false`

Checking Numeric Ranges with Logical Operators

- Used to test to see if a value falls **inside** a range:

```
if (grade >= 0 && grade <= 100)
  cout << "Valid grade";
```

- Can also test to see if value falls **outside** of range:

```
if (grade <= 0 || grade >= 100)
  cout << "Invalid grade";
```

- Cannot use mathematical notation:

```
if (0 <= grade <= 100) //doesn't work!
```

Validating User Input

- Input validation: inspecting input data to determine whether it is acceptable
- Bad output will be produced from bad input
- Can perform various tests:
  - Range
  - Reasonableness
  - Divide by zero

Validating User Input

```
16    int testScore;  // To hold a numeric test score
17
18    // Get the numeric test score.
19    cout << "Enter your numeric test score and I will\n"
20         << "tell you the letter grade you earned: ";
21    cin >> testScore;
22
23    // Validate the input and determine the grade.
24    if (testScore >= MIN_SCORE && testScore <= MAX_SCORE)
25    {
26       // Determine the letter grade.
27       if (testScore >= A_SCORE)
28         cout << "Your grade is A.\n";
29       else if (testScore >= B_SCORE)
30         cout << "Your grade is B.\n";
31       else if (testScore >= C_SCORE)
32         cout << "Your grade is C.\n";
33       else if (testScore >= D_SCORE)
34         cout << "Your grade is D.\n";
35       else
36         cout << "Your grade is F.\n";
37    }
38    else
39    {
40       // An invalid score was entered.
```

Comparing Characters

- Characters are compared using their ASCII values
- `'A' < 'B'`: The ASCII value of 'A' (65) is less than the ASCII value of 'B'(66)
- `'1' < '2'`: The ASCII value of '1' (49) is less than the ASCII value of '2' (50)
- Lowercase letters have higher ASCII codes than uppercase letters, so `'a' > 'Z'`

ASCII table

32: SPC	33: !	34: "	35: #	36: $
37: %	38: &	39: '	40: (	41:)
42: *	43: +	44: ,	45: -	46: .
47: /	48: 0	49: 1	50: 2	51: 3
52: 4	53: 5	54: 6	55: 7	56: 8
57: 9	58: :	59: ;	60: <	61: =
62: >	63: ?	64: @	65: A	66: B
67: C	68: D	69: E	70: F	71: G
72: H	73: I	74: J	75: K	76: L
77: M	78: N	79: O	80: P	81: Q
82: R	83: S	84: T	85: U	86: V
87: W	88: X	89: Y	90: Z	91: [
92: \	93:]	94: ^	95: _	96: `
97: a	98: b	99: c	100: d	101: e
102: f	103: g	104: h	105: i	106: j
107: k	108: l	109: m	110: n	111: o
112: p	113: q	114: r	115: s	116: t
117: u	118: v	119: w	120: x	121: y
122: z	123: {	124: |	125: }	126: ~
127: DEL				

Comparing Characters

```
10    // Get a character from the user.
11    cout << "Enter a digit or a letter: ";
12    ch = cin.get();
13
14    // Determine what the user entered.
15    if (ch >= '0' && ch <= '9')
16       cout << "You entered a digit.\n";
17    else if (ch >= 'A' && ch <= 'Z')
18       cout << "You entered an uppercase letter.\n";
19    else if (ch >= 'a' && ch <= 'z')
20       cout << "You entered a lowercase letter.\n";
21    else
22       cout << "That is not a digit or a letter.\n";
```

Comparing string Objects

- Like characters, strings are compared using their ASCII values

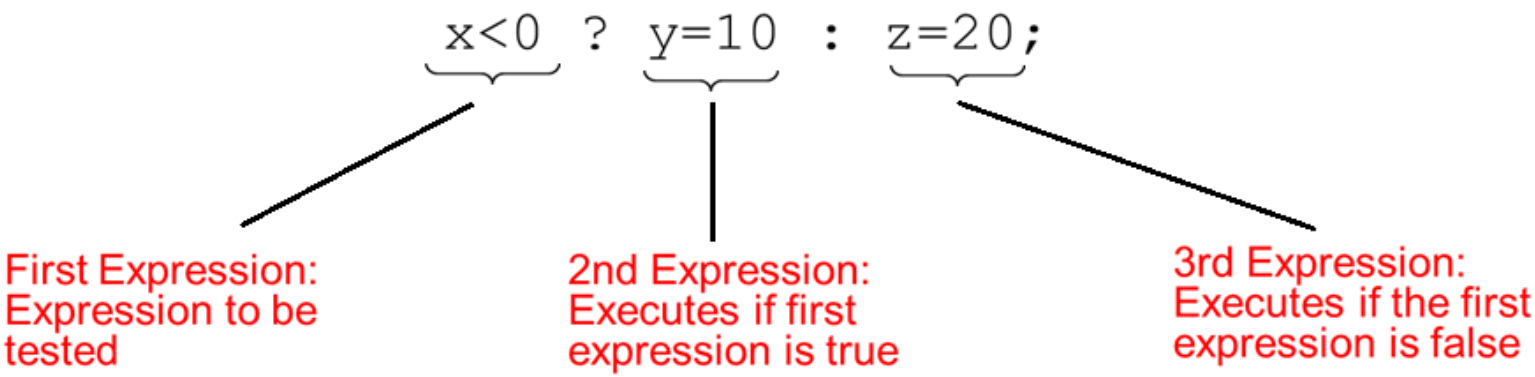
```
string name1 = "Mary";
string name2 = "Mark";

name1 > name2 // true
name1 <= name2 // false
name1 != name2 // true

name1 < "Mary Jane" // true
```

Comparing string Objects

```
26      // Determine and display the correct price
27      if (partNum == "S-29A")
28         cout << "The price is $" << PRICE_A << endl;
29      else if (partNum == "S-29B")
30         cout << "The price is $" << PRICE_B << endl;
31      else
32         cout << partNum << " is not a valid part number.\n";
```

The Conditional Operator

- Can use to create short `if/else` statements
- Format: `expr ? expr : expr;`



The Conditional Operator

- The value of a conditional expression is
  - The value of the second expression if the first expression is true
  - The value of the third expression if the first expression is false
- Parentheses () may be needed in an expression due to precedence of conditional operator

The Conditional Operator

```
1  // This program calculates a consultant's charges at $50
2  // per hour, for a minimum of 5 hours. The ?: operator
3  // adjusts hours to 5 if less than 5 hours were worked.
4  #include <iostream>
5  #include <iomanip>
6  using namespace std;
7
8  int main()
9  {
10    const double PAY_RATE = 50.0;   // Hourly pay rate
11    const int MIN_HOURS = 5;        // Minimum billable hours
12    double hours,                   // Hours worked
13           charges;                 // Total charges
14
15    // Get the hours worked.
16    cout << "How many hours were worked? ";
17    cin >> hours;
18
19    // Determine the hours to charge for.
20    hours = hours < MIN_HOURS ? MIN_HOURS : hours;
21
22    // Calculate and display the charges.
23    charges = PAY_RATE * hours;
24    cout << fixed << showpoint << setprecision(2)
25         << "The charges are $" << charges << endl;
26    return 0;
27 }
```

The switch Statement

- Used to select among statements from several alternatives
- In some cases, can be used instead of if/else if statements

```
switch (expression) //integer
{
  case exp1:  statement1;
  case exp2:  statement2;
  ...
  case expn:  statementn;
  default:  statementn+1;
}
```

`switch` Statement Requirements

- `expression` must be an integer variable or an expression that evaluates to an integer value
- `exp1` through `expn` must be constant integer expressions or literals, and must be **unique** in the `switch` statement

```
case (x < 20):  ....  // does not work
case (n == 20):  ....  // does not work
```

- `default` is optional but recommended

`switch` Statement-How it Works

1. `expression` is evaluated
2. The value of `expression` is compared against `exp1` through `expn`.
3. If expression matches value `expi`, the program branches to the statement following `expi` and continues to the end of the `switch`
4. If no matching value is found, the program branches to the statement after `default`

The switch Statement

Program 4-23

```
1  // The switch statement in this program tells the user something
2  // he or she already knows: the data just entered!
3  #include <iostream>
4  using namespace std;
5
6  int main()
7  {
8     char choice;
9
10    cout << "Enter A, B, or C: ";
11    cin >> choice;
12    switch (choice)
13    {
14       case 'A': cout << "You entered A.\n";
15                 break;
16       case 'B': cout << "You entered B.\n";
17                 break;
18       case 'C': cout << "You entered C.\n";
19                 break;
20       default:  cout << "You did not enter A, B, or C!\n";
21    }
22    return 0;
23 }
```

Program Output with Example Input Shown in Bold

```
Enter A, B, or C: B [Enter]
You entered B.
```

Program Output with Example Input Shown in Bold

`break` Statement

- Used to exit a `switch` statement
- If it is left out, the program "falls through" the remaining statements in the `switch` statement

The switch Statement

Program 4-25

```
1  // This program is carefully constructed to use the "fall through"
2  // feature of the switch statement.
3  #include <iostream>
4  using namespace std;
5
6  int main()
7  {
8     int modelNum;  // Model number
9
10    // Get a model number from the user.
11    cout << "Our TVs come in three models:\n";
12    cout << "The 100, 200, and 300. Which do you want? ";
13    cin >> modelNum;
14
15    // Display the model's features.
16    cout << "That model has the following features:\n";
17    switch (modelNum)
18    {
19       case 300: cout << "\tPicture-in-a-picture.\n";
20       case 200: cout << "\tStereo sound.\n";
21       case 100: cout << "\tRemote control.\n";
22                 break;
23       default:  cout << "You can only choose the 100,";
24                 cout << "200, or 300.\n";
25    }
26    return 0;
27 }
```

The switch Statement

Program Output with Example Input Shown in Bold

```
Our TVs come in three models:
The 100, 200, and 300. Which do you want? 100 [Enter]
That model has the following features:
    Remote control.
```

Program Output with Example Input Shown in Bold

```
Our TVs come in three models:
The 100, 200, and 300. Which do you want? 200 [Enter]
That model has the following features:
    Stereo sound.
    Remote control.
```

Program Output with Example Input Shown in Bold

```
Our TVs come in three models:
The 100, 200, and 300. Which do you want? 300 [Enter]
That model has the following features:
    Picture-in-a-picture.
    Stereo sound.
    Remote control.
```

Program Output with Example Input Shown in Bold

```
Our TVs come in three models:
The 100, 200, and 300. Which do you want? 500 [Enter]
That model has the following features:
You can only choose the 100, 200, or 300.
```

Using `switch` in Menu Systems

- `switch` statement is a natural choice for menu-driven program:
  - display the menu
  - then, get the user's menu selection
  - use user input as `expression` in `switch` statement
  - use menu choices as `expr` in `case` statements

More About Blocks and Scope

- Scope of a variable is the block in which it is defined, from the point of definition to the end of the block
- Usually defined at beginning of function
- May be defined close to first use

More About Blocks and Scope

```
16    if (income >= MIN_INCOME)
17    {
18       // Get the number of years at the current job.
19       cout << "How many years have you worked at "
20            << "your current job? ";
21       int years;     // Variable definition
22       cin >> years;
23
24       if (years > MIN_YEARS)
25          cout << "You qualify.\n";
26       else
27       {
28          cout << "You must have been employed for\n"
29               << "more than " << MIN_YEARS
30               << " years to qualify.\n";
31       }
32    }
```

Variables with the Same Name

- Variables defined inside { } have local or block scope
- When inside a block within another block, can define variables with the same name as in the outer block.
  - When in inner block, outer definition is not available
  - Not a good idea

Variables with the Same Name

Program 4-30

```
1  // This program uses two variables with the name number.
2  #include <iostream>
3  using namespace std;
4
5  int main()
6  {
7     // Define a variable named number.
8     int number;
9
10    cout << "Enter a number greater than 0: ";
11    cin >> number;
12    if (number > 0)
13    {
14       int number;  // Another variable named number.
15       cout << "Now enter another number: ";
16       cin >> number;
17       cout << "The second number you entered was "
18            << number << endl;
19    }
20    cout << "Your first number was " << number << endl;
21    return 0;
22 }
```

Program Output with Example Input Shown in Bold

```
Enter a number greater than 0: 2 [Enter]
Now enter another number: 7 [Enter]
```

Exercises

- Programming challenges: 13 - 18
- A program that asks users to enter two numbers (a and b) and an operator (+, -, *, /), then prints the result of applying that operator to the two numbers.
- A program that asks users to enter 4 numbers (a, b, c, d) and uses conditional operators to find the maximum of them.
- Write a program that asks the user to enter a two-digit integer and outputs the number in Vietnamese words.
- A program that asks users to enter an integer and outputs the hundreds digit of that number. If there is no hundreds digit, output 0.
- A program that asks the user to enter any valid date (day, month) in 2025 and determines what day of the week it is.