Computer Programming Using C
COP 3275 - Summer 2017

Lecture 7: Selection Statements

By Ahmed E. Khaled

/* Programming */
Recap to previous lecture!

- Logical Expressions
- Relational Operators
- Equality Operators
- Logical Operators
Logical Expressions

• The “Boolean” or “Logical” type would have only two values:
  1. true (integer value 1)
  2. false (integer value 0)

• There exist a set of operators that are used to build such logical expressions.
Relational Operators

- C’s *relational operators*:

<table>
<thead>
<tr>
<th>Symbol</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>&lt;</td>
<td>Less than</td>
</tr>
<tr>
<td>&gt;</td>
<td>Greater than</td>
</tr>
<tr>
<td>&lt;=</td>
<td>Less than or equal to</td>
</tr>
<tr>
<td>&gt;=</td>
<td>Greater than or equal to</td>
</tr>
</tbody>
</table>
# Equality Operators

- C provides two *equality operators*:

<table>
<thead>
<tr>
<th>Symbol</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>==</td>
<td>Equal to</td>
</tr>
<tr>
<td>!=</td>
<td>Not equal to</td>
</tr>
</tbody>
</table>
Logical Operators

• More complicated logical expressions can be built from simpler ones by using the **logical operators**:

<table>
<thead>
<tr>
<th>Symbol</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>!</td>
<td>Logical negation</td>
</tr>
<tr>
<td>&amp;&amp;</td>
<td>Logical and</td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Selection Statements in C
The `if` Statement

- The `if` statement allows a program to choose between *two alternatives* by testing an expression.
- In its simplest form, the `if` statement has the form

```plaintext
if ( expression )
{
    Statements
}
```
The if Statement

- When the expression is true, the statements are executed.

```c
int val = 70;
int min = 50;
int max = 90;
if (( min <= val ) && ( val <= max ))
{
    printf("Value %d is between %d and %d",
            val, min, max);
}
```
The **else** Clause

- An if statement *may have an else* clause:

```plaintext
if ( expression )
{
    Statements
}
else
{
    Statements
}
```

- The statement that follows the word `else` is executed *if the expression has the value false*. 
#include <stdio.h>

main() {
    int value = 600;
    int max = 500;

    if(value < max) {
        printf("value is less than the max");
    }
    else {
        printf("value is higher than the max");
    }
}
Nested If Else Clause

- It’s not unusual for if statements to be nested inside other if statements:

```c
if (i > j) {
    if (i > k) {
        max = i;
    } else {
        max = k;
    }
} else {
    if (j > k) {
        max = j;
    }
}
```
The “Dangling else” Problem

• When if statements are nested, the “dangling else” problem may occur:
  ```c
  if (y != 0)
    if (x != 0)
      result = x / y;
  else
    printf("Error: y is equal to 0\n");
  ```
• The indentation suggests that the else clause belongs to the outer if statement.
• However, C follows the rule that an else clause belongs to the nearest if statement that hasn’t already been paired with an else.
The “Dangling else” Problem

• A correctly indented version would look like this:

```c
if (y != 0)
  if (x != 0)
    result = x / y;
else
  printf("Error: y is equal to 0\n");
```
The “Dangling else” Problem

• To make the else clause part of the outer if statement, we can enclose the inner if statement in braces:

```c
if (y != 0) {
    if (x != 0)
        result = x / y;
} else
    printf("Error: y is equal to 0\n");
```

• Using braces in the original if statement would have avoided the problem in the first place.
Cascaded if Statements

- A “cascaded” if statement is used to write a series of test conditions, stopping as soon as one of them is true.

```plaintext
if ( expression )
  statement
else if ( expression )
  statement
...
else if ( expression )
  statement
else
  statement
```
Cascaded if Statements

• Example:

```c
if ( n < 0 )
    printf("n is less than 0 \n");
else if ( 30 < N )
    printf("n is equal to 0 \n");
else
    printf("n is greater than 0 \n");
```
Program: Calculating a Broker’s Commission

• When stocks are sold, the commission depends upon the value of the stocks traded shown in the following table:

<table>
<thead>
<tr>
<th>Transaction size</th>
<th>Commission rate</th>
</tr>
</thead>
<tbody>
<tr>
<td>Under $2,500</td>
<td>$30 + 1.7%</td>
</tr>
<tr>
<td>$2,500–$6,250</td>
<td>$56 + 0.66%</td>
</tr>
<tr>
<td>$6,250–$20,000</td>
<td>$76 + 0.34%</td>
</tr>
<tr>
<td>$20,000–$50,000</td>
<td>$100 + 0.22%</td>
</tr>
<tr>
<td>Over $50,000</td>
<td>$255 + 0.09%</td>
</tr>
</tbody>
</table>

• The minimum commission is $39.

• Develop program that asks the user to enter the amount of the trade, then displays the amount of the commission.
#include <stdio.h>

main()
{
    float commission, value;
    printf("Enter value of trade: ");
    scanf("%f", &value);

    if (value < 2500.00f)
        commission = 30.00f + .017f * value;
    else if (value < 6250.00f)
        commission = 56.00f + .0066f * value;
    else if (value < 20000.00f)
        commission = 76.00f + .0034f * value;
    else if (value < 50000.00f)
        commission = 100.00f + .0022f * value;
    else
        commission = 255.00f + .0009f * value;

    if (commission < 39.00f)
        commission = 39.00f;

    printf("Commission: $%.2f\n", commission);
}

Write a program

• Your program determines the number of digits in a number.

• Assume that the number has no more than four digits.

• Test case of your program run:

  Enter a number: 342

  The number 342 has 3 digits
Write a program

• Your program asks the user for a 24-hour time, then displays the time in 12-hour form.

• 0:00 and 12:00 are special cases

• Test case of your program run:

  *Enter a 24-hour time: 21:11*

  *Equivalent 12-hour time: 9:11 PM*