Conditional Statements in C

Conditional Statements in C

Sequential Programming Limitation

Sequential Programming
- Computer executes instructions in the order in which they are stated

Limitations of Sequential Programming
- Cannot choose whether or not to perform a command/instruction
- Cannot perform the same command more than once

Control Structures

Conditional
- Making decision about which code to execute, based on evaluated expression
  - if
  - if-else

Iteration
- Executing code multiple times, ending based on evaluated expression
  - while
  - for
  - do-while

"if" statement

if (condition){
    statement(s)
}

If the condition is true (expression is evaluated to non-zero), then the statement(s) (i.e. instructions) will be executed. Otherwise, it/they won’t.

//Assume x is an integer
if(x > 10) {
    x = x * 2;
}
“if” statement (contd..)

{} indicates the block of code that will get executed given the condition is true
You can avoid the curly brace after condition if only one statement is to be performed

```c
//Assume x is an integer
if(x < 0)
    x = x * -1;
```

Example if Statements

```c
if (x <= 10)
    y = x * x + 5;
```

- Style: avoid singleton if statements (Often cause of errors)

```c
//Assume x is an integer
if(x > 0) {
    x = x * 2;
} else {
    x = x * -1;
}
```

only first statement is conditional; second statement is always executed

```
if (x <= 10) {
    y = x * x + 5;
    z = (2 * y) / 3;
}
```

- compound statement; both executed if x <= 10

Assignment vs. Equality

Don’t confuse equality (==) with assignment (=)

```c
int x = 9;
int y = 10;
if (x == y) {
    printf("not executed\n");
}
if (x = y) {
    printf("x = %d y = %d", x, y);
}
```

Result: “x = 10 y = 10” is printed. Why?

Compiler will not stop you! (What happens in Java?)

```
if (condition){
    statement(s)
}
else{
    statement(s)
}
```

```
//Assume x is an integer
if(x > 0) {
    x = x * 2;
} else {
    x = x * -1;
}
```
If-else Flow chart

true

condition?

false

Statement(s)

Statement(s)

Style Rule: Indentation and Spacing

- Recommended indentation is from 2 to 4 spaces, but must be consistent throughout the program
- Single space around every binary operator, including comparisons and assignment (=)

```c
if (x < 10) {
    x = x + 1;
} else {
    x = x - 1;
}
```

Cascading “if-else” Example

```c
// Assume variable score is entered by user
if (score > 90)
    printf("Grade A\n");
else if (score > 80)
    printf("Grade B\n");
else if (score > 65)
    printf("Grade C\n");
else
    printf("F\n");

// Note: You can avoid the curly brace after condition if only one statement is to be performed
```

Cascading/Chaining If’s and Else’s Example 2

```c
if(month == 4 || month == 6 || month == 9 || month == 11) {
    printf("Month has 30 days.\n");
}
else if (month == 1 || month == 3 || month == 5 || month == 7 || month == 8 || month == 10 || month == 12) {
    printf("Month has 31 days.\n");
}
else if (month == 2) {
    printf("Month has 28 or 29 days.\n");
}
else {
    printf("Don’t know that month.\n");
}
```
**Difference between Chaining if-else vs. just using if**

```java
int x = -5;
if(x < 0){
    x = -x;
} else if(x >= 1 && x <= 10){
    x = x * 2;
} else if(x > 10){
    x = x * 2 - 1;
}
```

If a series of standard "if" statements were used, it would be possible for all of the statements to be executed, which is not what was intended.

**Nested if-statements**

<table>
<thead>
<tr>
<th>A</th>
<th>B</th>
<th>C</th>
</tr>
</thead>
<tbody>
<tr>
<td>T</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

What values must the conditions have in order for block A to run? B? C?

**Matching Else with If**

Else is always associated with closest unassociated if

```java
if (x != 10) {
    if (y > 3)
        z = z / 2;
    else
        z = z * 2;
}
```

is NOT the same as...

```java
if (x != 10) {
    if (y > 3)
        z = z / 2;
} else
    z = z * 2;
```

**Switch**

```java
switch (integer_expression){
    case const1:
        statement(s);
        break;
    case const2:
        statement(s);
        break;
    default:
        statement(s);
        break;
}
```

Alternative to long if-else chain. If break is not used, then case "falls through" to the next.
/** same as month example from cascaded if-else */
switch (month) {
    case 4: case 6: case 9: case 11:
        printf("Month has 30 days.\n");
        break;
    case 1: case 3:
        /* some cases omitted for brevity...*/
        printf("Month has 31 days.\n");
        break;
    case 2:
        printf("Month has 28 or 29 days.\n");
        break;
    default:
        printf("Don’t know that month.\n");
}