Branching
Announcements

Homework 1 out!
Homework 0 due tomorrow at 5:30
(make sure you read instructions carefully)

Make sure your lab scores are correct (email a TA or me if not)
Learning to talk to the computer (programming) is half the battle (Ch. 2 stuff)

The other half is formalizing the solution as directions, which we will do a bit today
Outline

Last time (+,=,<, int, double, char)

if/else statements cont.
boolean expressions
Nested if/else
Multi-way if/else (switch)
Scope
bool

bool - either true or false

C++ lets you change between fundamental types (casting) with ease

Q: 0 is false and 1 is true, right?
A: 1.
Sometimes this might cause an error, such as:

```java
int x = 7;
if( ! x>5 ) will be false
```

Why?
boolean values

Sometimes this might cause an error, such as:

```java
int x = 7;
if(! x>5 ) will be false
```

Why?
A: order of operations will do the unrary operator first (the '!' )
if (! x>5) will become if ( (!7) > 5)
... if ( (!true) > 5) ... if ( false > 5) ... if (0 > 5)
if statement

Code inside an **if statement** is only run if the condition is true.

Need parenthesis (no semi-colon)

```cpp
if(guess == random0to9)
{
    cout << "Correct, here is a cookie!\n";
}
```

Indent

(See last week:  numberGuessing.cpp)
if/else statement

Immediately after an if statement, you can make an else statement.

If the “if statement” does not run, then the else statement will.

If you do not surround your code with braces, only one line will be in the if (and/or else) statement.
if/else statement

(See: ATM.cpp)
if/else statement

Write an if statement for checking if a variable (int) x is a positive odd number.

Hint: You may want to use the remainder (also called modulus) operator (the % sign).

For example, 5 % 3 = 2
Logical operators

These are all the operators that result in a `bool`:

> (greater than), e.g. 7 > 2.5 is true

== (equals), e.g. 5 == 4 is false

< (less than), e.g. 1 < 1 is false

>= (greater than or equal to), e.g. 1 <= 1 is true

!= (not equal to), e.g. 8 != 7 is true

<= (less than or equal to), e.g. 6 <= 2 is false

! (not, negation), e.g. !true is false
Complex expressions

Two boolean operators:
&& is the AND operations
|| is the OR operations
Complex expressions

AND operation removes Ts from the result
The OR operation adds Ts to the result

Evaluate (!p OR q) AND (p)
Complex expressions

Humans tend to use the English word OR to describe XOR (exclusive or).

“We can have our final exam on the scheduled day (Dec. 22) or the last day of class (Dec. 15).”

Did you think the statement above meant final exams on both days was a possibility?
Complex expressions

Write boolean expressions for each of the following truth tables:

<table>
<thead>
<tr>
<th>A</th>
<th>B</th>
<th>Out</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>0</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>A</th>
<th>B</th>
<th>Out</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>A</th>
<th>B</th>
<th>Out</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>0</td>
</tr>
</tbody>
</table>

XOR
Complex expressions

```java
int x = 9, y = 7;
```

\[
x < 12 \land \land y > 10
\]
\[
9 < 12 \land \land 7 > 10
\]
\[
T \land \land F
\]
\[
F
\]
Complex expressions

If statements for when $x$...

... is between 10 and 20 (inclusive)

```java
if(10 <= x && x <= 20)
```

Cannot say: $10 \leq x \leq 20$ (why?)

... is a vowel ($x$ is type `char`)

```java
if( x == 'a' || x == 'e' || x == 'i' || x == 'o' || x == 'u')
```
Complex expressions

Write a single if-statement that is true on the following range of numbers:
sample) \textbf{int} i: 3

Answer: if( i == 3)

a) \textbf{int} i: \ldots -2, -1, 0
b) \textbf{int} i: 5, 6, 7, 8, \ldots
c) \textbf{int} i: 1, 2, 3, 4, 5
d) \textbf{int} i: \ldots -2, -1, 1, 2, 3, \ldots
e) \textbf{int} i: \ldots -2, -1, 5, 6, 7, \ldots
Be careful when negating, that you follow De Morgan's Law:

```cpp
bool a, b;
!(a OR b) is equivalent to (!a) AND (!b)
!(a AND b) is equivalent to (!a) OR (!b)
```

“Neither rainy or sunny” means
“Both not rain and not sunny”
bool life = false;
bool money = false;
Nested if statements

You can have as many if statements inside each other as you want.

```java
if (teacherAwake)
{
    if (studentAwake)
    {
        if (classWellPrepared)
        {
            learning = true;
        }
    }
}
```
Nested if statements

From a truth table perspective, nested loops are similar to AND

The previous if code is equivalent to:

```java
if (teacherAwake && studentAwake && classWellPrepared) {
    learning = true;
}
```

However, sometimes you want to do other code between these evaluations.
Nested if statements

(See: bridgeOfDeath.cpp)
Scope

Where a variable is visible is called its **scope**

Typically variables only live inside the block (denoted with matching `{` and `}`)

A variable lives until the block is closed, so inner blocks can see everything from the block it was created inside
Scope

```cpp
int main()
{
    int x;
    // can use x here
    {
        int y;
        // can use x or y here
    }
    // can use x here
    return 0;
}
```

(See: scope.cpp)
If... if... else!

When in doubt, use parenthesis and blocks! (Some people like to put the first brace after the if, others on a new line)

What happens if you have an if if else?

(See: ifIfElse.cpp)
Multiway if/else

This is a special format if you put an if statement after an else.

This second “if statement” only is tested when the first “if statement” is not true

(See: grades.cpp)
Multiway if/else

(See: vending.cpp)
Switch

A switch statement checks to see if a variable has a specific value.

```cpp
switch( controllingVariable )
{
    case 2:
    case 4:
        cout << "controllingVariable is either 2 or 4" << endl;
        break;
    case 3:
        cout << "controllingVariable is 3\n";
        break;
    default:
        cout << "controllingVariable is not 2, 3 or 4...\n";
        break;
}
```
Switch

If the value of the controlling variable is found in a case label, all code until a break statement is ran (or the switch ends)

Switch statements only test equality with case labels (not greater or less than)

(See: switch.cpp)
Switch

Switch statements can be written as multiway if/else statements.

Could use just “if statements” but “else if” shows only one of these will run

(See: switchToIf.cpp)
Conditional operator

We will not use in this class, but if you use other people's code you will encounter

Shorthand for an if-else statement

(boolean) ? [if true] : [if false]

Example:
max = (x>y) ? x : y;
(See: max.cpp)
Football (American!)

(See: football.cpp)