Strings & Branching

Strings and input

We talked about basic types....
what type can store letters?

What about words?

Input and output
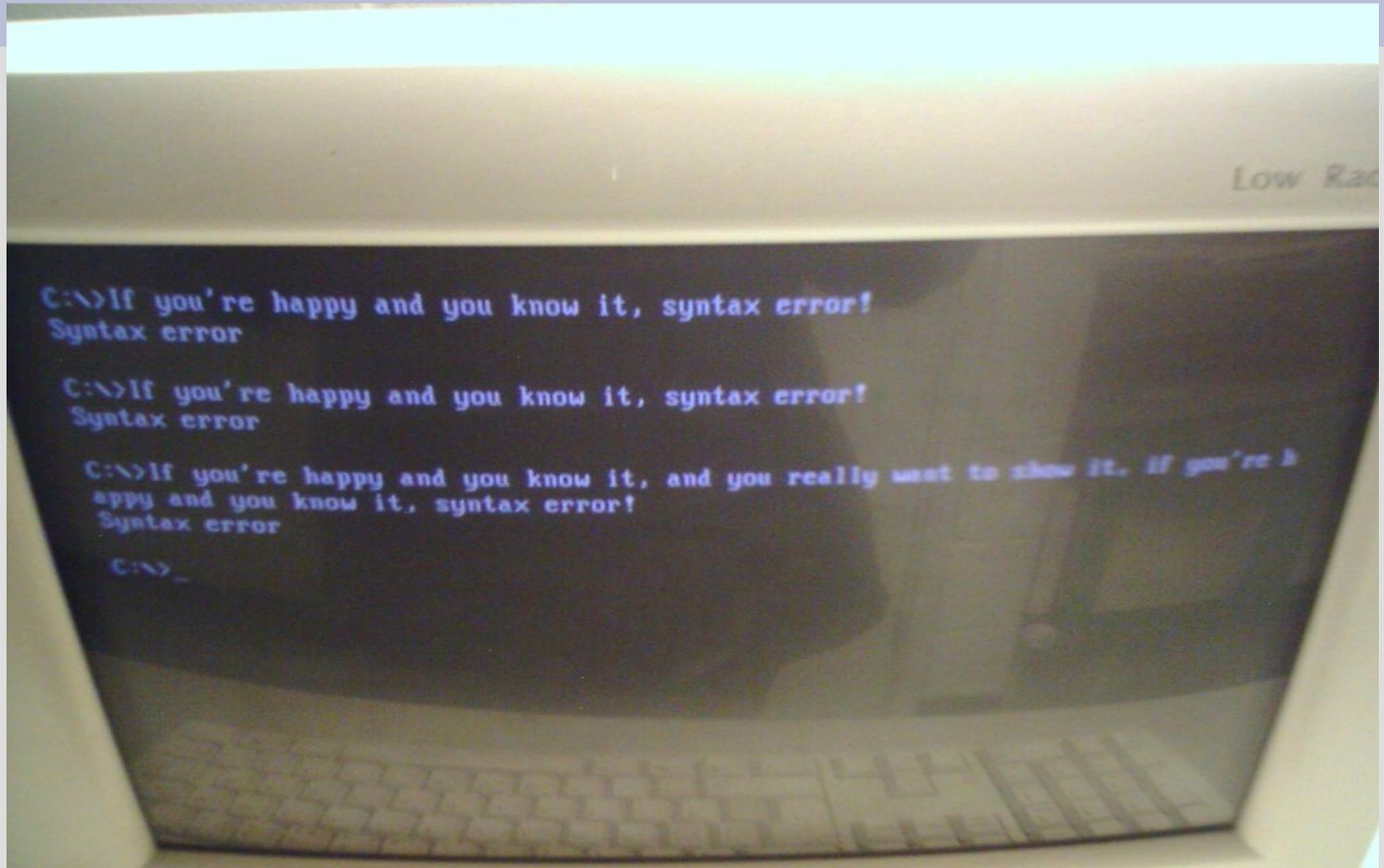
Strings and input

char can only hold a single letter/number, but one way to hold multiple is a string

string str;
cin >> str;

The above will only pull one word, to get all words (until enter key) use:

getline(cin, str);   (See: stringInput.cpp)

Miscellaneous cin info

With cin, it will stop as soon as it reaches a type that does not match the variable (into which it is storing)

If it encounters only a type that it is not expecting, your input will get messed up

cin also remembers all inputs
(See: cinMismatchTypes.cpp)

More Output

When showing doubles with cout, you can change how they are shown

For example, to show a number as dollars and cents, you would type (before cout):

```
cout.setf(ios::fixed);
cout.setf(ios::showpoint);
cout.precision(2);
```

More Output

There are two ways to get output to move down a line: endl and "\n"

    cout << endl;

... is the same as...

    cout << "\n"

I will use both when coding

Madlibs
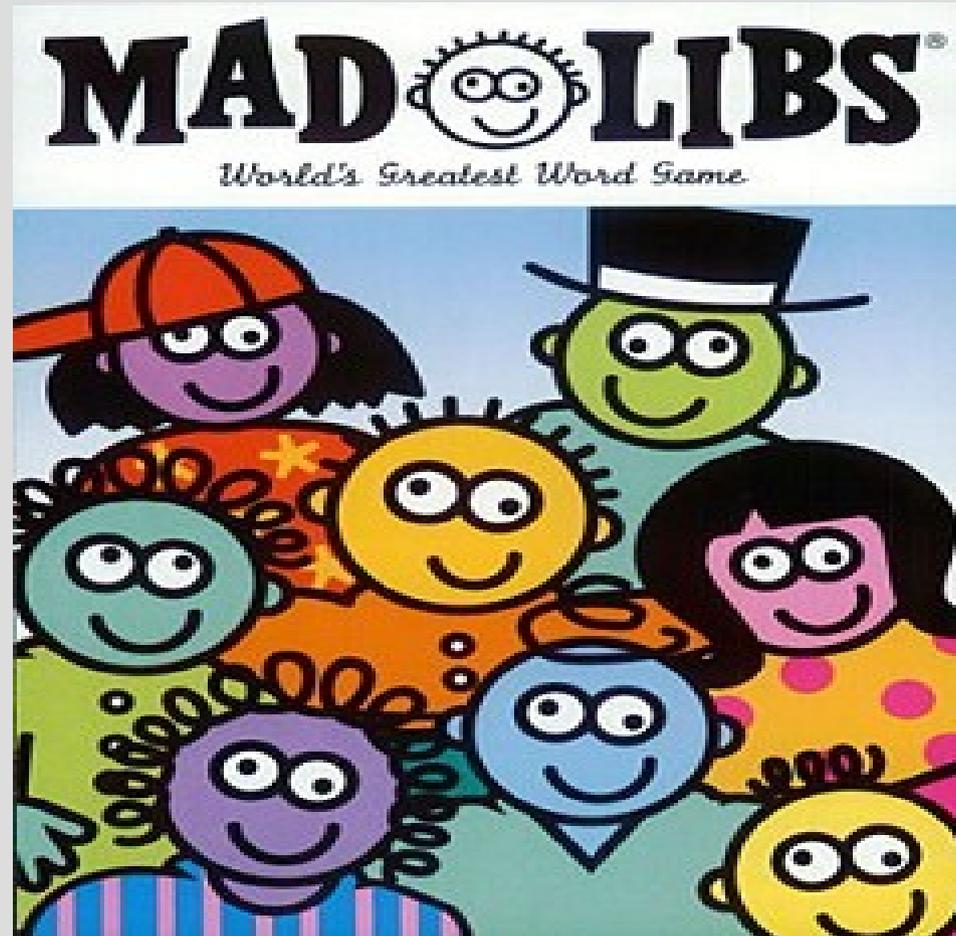
(see: madlibs.cpp)

bool

bool - either true or false

You have the common math comparisons:
   > (greater than), e.g. 7 > 2.5 is true
   == (equals), e.g. 5 == 4 is false
   <= (less than or eq), e.g. 1 <= 1 is true

If you cout this, "false" will be 0
and "true" will be 1 (anything non-zero is T)

if statement

Code inside an <u>if statement</u> is only run if the condition is true.

Need parenthesis
(no semi-colon)

```cpp
12    if(guess == random0to9)
13    {
14        cout << "Correct, here is a cookie!\n";
15    }
```

Indent

(See: ifElse.cpp)

boolean values

ints will automatically be converted to bool, which can cause errors:

int x = 2;

if(! x>5) will be false

Why?

boolean values

ints will automatically be converted to bool, which can cause errors:

int x = 2;

if(! x>5) will be false

Why?

A: order of operations will do the unary operator first (the '!')

if (! x>5) will become if ((!2) > 5)

... if ((!true) > 5) ... if (false > 5) ... if (0 > 5)

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

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

p	q	p && q
T	T	T
T	F	F
F	T	F
F	F	F

p	q	p || q
T	T	T
T	F	T
F	T	T
F	F	F

Complex expressions

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

Evaluate (!p OR q) AND (p)

p	q	!p	!p OR q	(!p OR q) AND (p)
T	T	F	T	T
T	F	F	F	F
F	T	T	T	F
F	F	T	T	F

Complex expressions

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
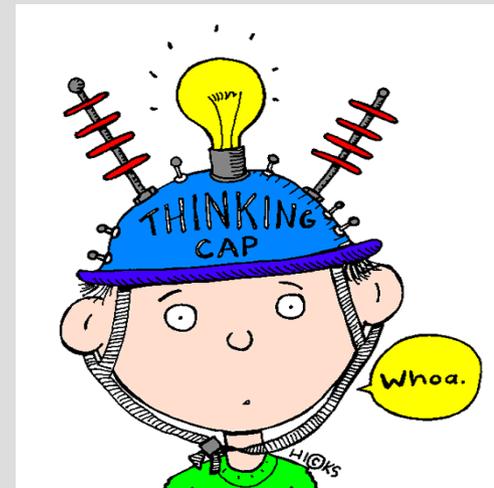
Complex expressions

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

"You can get a side of a salad, fries or a soup."

Did you think the statement above meant getting all three was a possibility?

Complex expressions

Write boolean expressions for each of the following truth tables:

1.

A	B	Out
0	0	0
0	1	0
1	0	0
1	1	0

2.

A	B	Out
0	0	0
0	1	0
1	0	1
1	1	1

3.

A	B	Out
0	0	0
0	1	0
1	0	1
1	1	0

4.

A	B	Out
0	0	0
0	1	1
1	0	1
1	1	0

XOR

Complex expressions

int x = 9, y = 7;

$$x < 12 \;\&\&\; y > 10$$

$$9 < 12 \;\&\&\; 7 > 10$$

$$T \;\&\&\; F$$

$$F$$
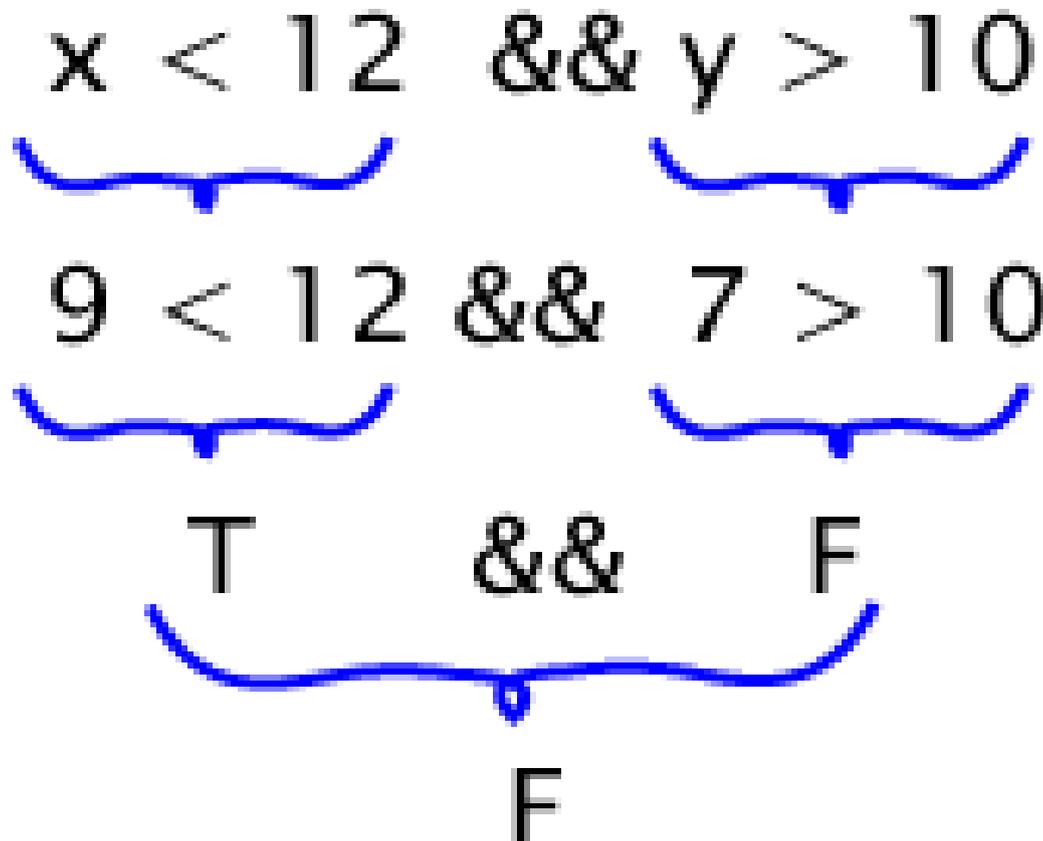
; and if

Please always put {} after if-statements

The compiler will let you get away with not putting these (this leads to another issue)

If you do not put {} immediately after an if, it will only associate the first command after with the if-statement
(see: ifAndSemi.cpp)

Random numbers

To use random numbers, you need to do:
    1. Run srand(time(0)) once
    2. Use rand() to actually generate a number

```cpp
int main()
{
    srand(time(0));

    cout << rand()%10 << endl; // displays 0-9
}
```

<u>DO ONLY ONCE AT THE START OF MAIN AND NEVER AGAIN!</u>

(See: rng.cpp)