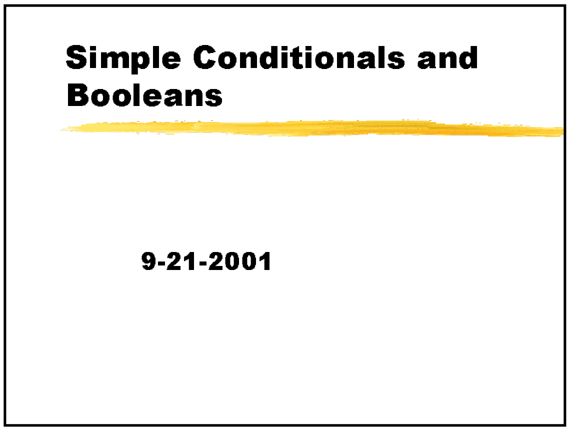
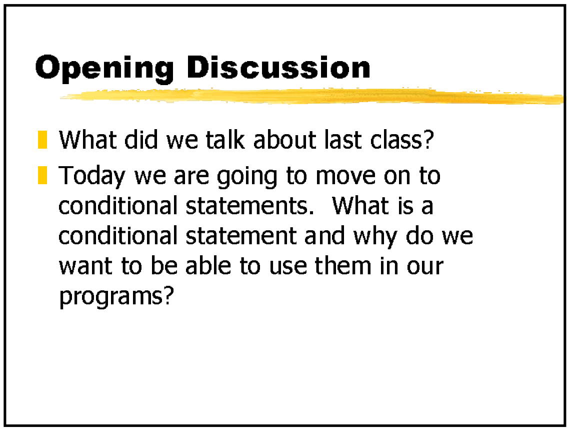


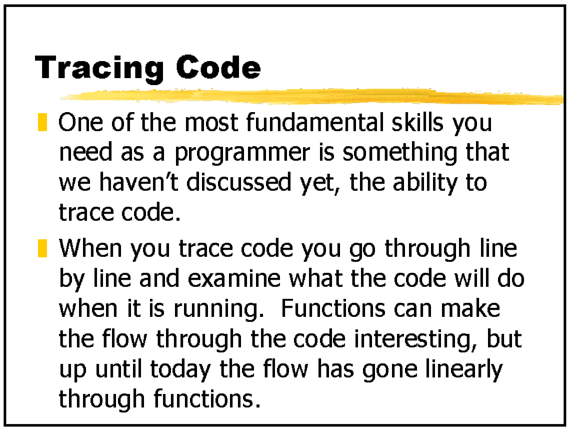
Simple Conditionals and Booleans

9-21-2001



Opening Discussion

- What did we talk about last class?
- Today we are going to move on to conditional statements. What is a conditional statement and why do we want to be able to use them in our programs?



Tracing Code

- One of the most fundamental skills you need as a programmer is something that we haven't discussed yet, the ability to trace code.
- When you trace code you go through line by line and examine what the code will do when it is running. Functions can make the flow through the code interesting, but up until today the flow has gone linearly through functions.

Function Call Expressions

- The fact that functions with return values are valid expressions means that they can be used any place an expression can. This implies that they can be used in arguments to other functions. This is standard and it appears in the ChangeMaker problem in the book.
- The function in the argument must be evaluated before the outer function can be called. At this stage it also needs to appear in an argument that is passed by value.

if/else Statements

- The basic conditional statement found in virtually every programming language is the if statement. Generally there is also the ability to have an "else" option.

```
if(condition) {
     // When condition is true execute
     // this block.
} else {
     // When condition is false execute
     // this block.
}
```

A Word on Style

- The placement of brackets in an if/else statement is a very subjective topic (it can be borderline religious at times). I use a brackets that save screen space keeping the open bracket on the same line as the if or else. Your book goes along the line that they should have their own line and be lined up. There are other conventions as well. I don't care what you use as long as there is some logic and you are consistent.
- The code in the block SHOULD be indented.

Boolean Expressions

- The condition in the code is generally what would be called a boolean expression. That is an expression that has a value of true or false.
- Boolean expressions can contain normal arithmetic operators but they typically contain other operators as well.

Comparison Operators

- The following are operators that you can use to compare numeric values in C++. They return boolean values.
- == - This is the equal sign in C++.
- <, > - Less than and greater than.
- <=, >= - Less than or equal to and greater than or equal to.
- != - Not equal to.

Boolean Operators

- There are other operators that operate on boolean values.
- ! - unary not operator
- && - and
- || - or - This is not the way we typically use or in English because it means one or the other or both.
- ^ - xor - This is what we use in English. xor stands for exclusive or.

Short-circuit Boolean Evaluation

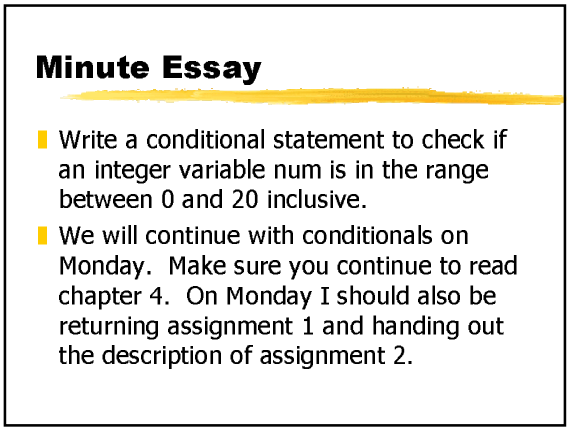
- If you recall I previously listed & and | as bitwise numeric operators. They actually work as boolean operators too, but the double versions are typically used instead because they are short-circuit operators.
- This means that if the value of the left argument determines the value of the whole expression the right argument isn't evaluated.

Integer Value of Booleans

- C++ does not have a specific type for booleans. There is a bool type but it is essentially an int.
- When an integer value is used in a conditional 0 is interpreted as false and anything else is interpreted as true.
- Unfortunately this implies that two expressions that evaluate as true in a conditional might not be equal to one another.

== is NOT =

- One of the most common mistakes in C++ is to put a single = in a check for equality where == should be used.
- Because boolean values are basically ints, this is generally not a syntax error though it is almost always a logic error.
- Our compiler does not flag this with a warning, even when you use the -pedantic flag.



Minute Essay

- Write a conditional statement to check if an integer variable num is in the range between 0 and 20 inclusive.
- We will continue with conditionals on Monday. Make sure you continue to read chapter 4. On Monday I should also be returning assignment 1 and handing out the description of assignment 2.