*Notes 10/6*

The “is equal to” operator usually gives trouble; two equal signs indicate to check whether 2 values are equal, while one assigns the right value to be equal to the left.

X **=** 8 🡪 makes X = 8.

X **==** 8 🡪 checks if X = 8.

Not equal to: **!=**.

Code fragment to check if no input was entered for a name:

*if (name == “”)  
 cout << “You didn’t type a name!” << endl;  
else  
 cout << “Hello, “ << name << endl;*

There’s an alternate form of the if statement that is similar to the *if/then* statements in Visual Basic.

*if (condition)  
 stmtIfTrue*

If the condition is false the program does nothing and ignores the code fragment.

Easy mistake to make: don’t put semicolons at the end of if statements or it will mess up your code. If stuff isn’t working and you can’t find the bug, this is the first place to check. Semicolons go at the end of the stmtIfTrue, not at the end of the list of conditions.

Declared variables are only relevant within the curly braces they are declared in. In addition, if you declare a variable in an if statement it will only be valid within the statement – so if you reference the variable later you will get a compiler error, even if you don’t put curly braces around your if statement conditions.

You can declare a variable outside a curly brace restriction and then initialize it within the statement; for example, you can declare variable withholdingRate outside of an if statement, and then assign it a value of .1 in the if conditions and .05 in the else conditions. This is valid because the variable is declared for use by both conditions, but the value assigned to it only holds within the scope of the individual conditions.

Not initializing integers or doubles gives them a random value if you reference them without initializing them fist (bad idea.) Strings, on the other hand, default to an empty string if you leave them uninitialized so it’s fine to leave them blank instead of initializing them as “”.

“Magic numbers” are numbers in your program that have a reasonable chance of being changed in further updates to the code down the line.

If you think you might have to change the value later on down the line, it’s better to declare the “magic number” as a variable right at the top of the program so if it ever has to be revised, finding and changing the value is simple.

Use the *const* modifier to initialize “magic numbers.” Using this in front of *double* or *int* “protects” the initialized variable from being changed in the program by accidentally putting = instead of ==. You basically promise to not ever change the variable again – the compiler will issue an error if you try to change it.

*Const* values go at the top of the code, right under *int main()* usually.

You can also use *const* modifiers to declare things like pi, e, etc. – even though these values won’t change, you should still declare them as constants instead of typing them out every time because it makes the code easier to read and easier to write.

The = sign does not mean what it means in math for C++. In C++ the statement *n = n+10* is valid, in math it is not. In C++ this simply means to add 10 to the value of n. *Whatever is on the left of the = sign is assigned the value of whatever is on the right of the = sign.*

Initialized variables are not reciprocal. For example:

*int a = 3;  
int b = a + 5; //* ***b is now 8***

*a = 4 //* ***b is still 8****b = a + 40 //****b is now 44***

This is the opposite of a spreadsheet program which retroactively changes declared values. C++ does not, don’t get confused.

You can nest if statements within if statements. Example:

*if (citizenship == “US”)  
{  
 if (age >= 18)  
 cout << “You can vote in US elections” << endl;  
}   
else cout << “You are not a US citizen” << endl;*

Be careful with if/else statements if you use nested if statements. Else statements automatically pair up with the lowest-level if statement, so if you just want an “if-then” type nested if statement protect it with curly braces so the “else” does not assign itself to it.

|| means “OR” in C++; && means “AND”.

*If (citizenship == “US” || citizenship == “Canada”)  
 cout << “The country code is 1” << endl;*

*If (citizenship == “US” && age >= 18)  
 cout << “You can vote in US elections” << endl;*