*Notes 10/11*

&& operators have higher precedence than || operators, but if you use both it is best to use parenthesis for ease of reading.

Make sure when you use < or > operators you check for values = to those you are comparing. For example if you say (if (cin > 50)), check your code for what happens if cin = 50.

When you use && or || operators you have to write out a complete sub expression on each side of the operator – you can’t shorten it like you do in English.

The opposite of <= is >, and the opposite of >= is <. Be very careful about using <= and >= referring to the same variable or the program might do both functions when the exact value is inputted.

De Morgan’s Laws:

*Not (A and B) 🡺 (not A) or (not B)  
Not (A or B) 🡺(not A) and (not B)*

Syntax *a < integer < b* is not valid in C++. You have to say *a < integer && integer < b.*

If ladders can be used with *if/else if/else* structures. You can use less conditionals and optimize the run speed of your code.

You can use a *switch* structure if you are referencing constants in your conditionals instead of an if ladder.

Syntax is

*Switch (choice)  
{  
 case 1:  
 …;  
 break;  
 case 2:  
 case 3:  
…  
 default:  
 …  
 return 1;  
}*To break out of the switch statement you must use *break* at the end of your case statement. It is kinda of unintuitive but if you forget to do it the switch will just keep on going through the cases until it reaches a break.

If none of the cases are true the switch statement will run the *default* you define at the end. Don’t forget a default, usually an error message, to prevent the program from crashing.

You can’t use a switch statement for strings (annoying because it would be quite useful.)

*While* statements initiate loops. Loops are useful (at a basic level) for having a computer perform some task N number of times, with N inputted by the user.

An example loop would be

*Int nTimes = 0  
Cout >> “How many times would you like to be greeted? “;  
Cin << nTimes  
Int n = 1;  
While (n <= nTimes)  
{  
 cout << “Hello” << endl;  
 n += 1  
}*

This loop would display “Hello” the number of times requested by the user using a *while* loop.

The symptom of using < when you should have used <= or vice versa in a loop is when the loop goes through one time too many or one time to few.

(N += 7 is the same as N = N + 7.) Other ones are \*=, /=, -=, etc.

The increment operater is n++. It means N = N + 1. Decrement is n--.

If you want to add 1 to n or subtract 1 from n in a condition, before taking the value of the number, you can do ++n or - -n.

**Loops and Strings on the CS31 website has good info about help on project 3.**