Decisions

In this section, we're going to learn how to control the flow of a C++ program. Sometimes, you want your program to do one thing if certain conditions are met, and to do another thing if other conditions are true. We do this with if-statements.

Let's imagine we have a program that helps police officers in a town give speeding tickets to drivers based on how fast the driver was going. In this imaginary town, the speed limit is 40 mph, but there are extra penalties if a person goes over 60 mph.

Here's how the program will work: the policeman (the user) enters the speed of a vehicle in the program. If the vehicle was going over 40 mph, the driver gets a $100 speeding ticket. But, if the vehicle was going over 60 mph, the driver gets charged $250. If the vehicle was going 40 mph or under, the driver doesn't get a ticket. The program will display the fine to the user.

In order to do this, we will need to test the speed that the user types in (we'll save this to a variable called *driverSpeed* and see if it is greater than 40, and then see if it is also greater than 60. Just like we would in math, we can write this expression using the greater than/less than symbols and store it in a variable:

```
isSpeeding = driverSpeed > 40;
```

In programming languages, the above statement can either be true or false, depending upon the value of *driverSpeed*, so *isSpeeding* will either be true or false. A variable that is either true or false is called a boolean. In C++ , a boolean that is true displays as a 1, and a boolean that is false displays as a 0 when you print it out to the screen. So, in our line of code above, if the value of the variable *isSpeeding* is 20, and you printed out *isSpeeding*, you would see a 0, because 20 is not greater than 40 and the statement is false.

The following chart shows all the logical operators, most of which should be familiar to you from math class:

Logical Order	Meaning	Example of True Statement
>	greater than	`5 > 4`
>=	greater than or equal to	`5 >= 5`
<	less than	`2 < 5`
<=	less than or equal to	`2 <= 3`
==	exactly the same	`3 == 3`
!=	not the same	`3 != 5`

The last two operations, == and !=, are called equivalency operators because they check if two things are exactly the same, or equivalent. For example, 9 == 9 is true because 9 is exactly the same as 9. Additionally, if we have a variable *myVar* and *myVar* = 9, then *myVar* == 9 is true. However, 9 != "nine" is true because an integer (9) is never exactly the same as a string ("nine"). One more important thing to remember about strings is that capital letters and lowercase letters are considered to be different, so "Nine" and "nine" are not the same.

In order to test things, we can use if-statements. An if-statement is used in programming exactly the same way we use it in real life. "If" something is true, we do one thing; "else" (otherwise) we do something else. To come back to the speeding ticket program we described above, we could test whether the variable *isSpeeding* is greater than 40 by using the following syntax:

```
if (isSpeeding > 60){
      ticketPrice = 250;
}
else {
      ticketPrice = 0;
}
```

When we use if statements, we put the conditional clause (the thing we are checking or testing) in parentheses after the word "if". Then we put all of the code that we want the computer to execute if the conditional clause is true inside of curly braces. After an "if", we can have an "else if ", which works exactly the same as an "if" but is only checked by the computer if the "if" statement is false, an else, which is the code that is executed if the "if" statement is false, or nothing.

We now know everything we need to be able to write the speeding ticket program. We will use an if-else pattern for this program, so we will first check if the speed is greater than 60, and if that is false we will check if the speed is greater than 40, and if that is also false, then we will give a $0 speeding ticket.

```
#include <iostream>
using namespace std;

int main(){

      int driverSpeed, ticketPrice;

      cout << "Hello policeman.  Enter the speed of the vehicle:";
      cin >> driverSpeed;

      if (driverSpeed > 60){  //First check if driver is excessively speeding
            ticketPrice = 250;
      }
      else if (driverSpeed > 40) {//Then check if driver was going over 40mph
            ticketPrice = 100;
      }
      else {      //If neither of those were true, the driver wasn't speeding
            ticketPrice = 0;
      }

      cout << "The ticket is $" << ticketPrice << ".\n";
}
```

Challenge 1:

Can you modify the above code so that it prints out a separate message for each different level of ticket? For example, if the driver was going over 60mph, the program could print "You should severely punish the driver for endangering people!" and for a non-speeding ticket, the program could print, "Congratulate the driver on maintaining a safe speed."

You can also nest if-statements inside each other. For example, let's say that there is an additional law about wearing seat belts in our imaginary town. If you are speeding and you are also not wearing a seat belt, you get fined an additional $55. We can write add this into our code like this:

```
[...] // ignoring the beginning stuff

int driverSpeed, ticketPrice;
string wearingSeatbelt;     //string variable

//get speed
cout << "Hello policeman.  Enter the speed of the vehicle:";
cin >> driverSpeed;

//user will type y or n
cout << "Was the driver wearing a seat belt? y or n:";
cin >> wearingSeatbelt;

//First check if driver is excessively speeding
if (driverSpeed > 60){
   //set the initial ticket price to 250
      ticketPrice = 250;

      //if the speed is over 60 AND
      //the driver wasn't wearing a seat belt
      if (wearingSeatbelt == "n"){
            //add 55 to the old ticket price (250+55=305)
            ticketPrice = ticketPrice + 55;
      }
}
[...]  //rest of the program
```

You can also check if two or more conditions are both true in the same if-statement. You do this by using &&. So you can say,

```
//true if both conditionals are true
if ((driverSpeed > 60) && (wearingSeatbelt == "n")){
           ticketPrice = 305;  //250 + 55
}
```

If BOTH the first clause (driverSpeed > 60) and the second clause (wearingSeatbelt == "n") are true, then the whole conditional is true. If either are false, the whole thing is false. There is also an "or" operator ||, which is true if either or both of the conditions are true:

```
// true if one or both conditionals are true
if ((driverSpeed > 60) || (wearingSeatbelt == "n")){
           cout << "You broke the law, buddy!\n";
}
```

Note how we use parentheses to group each conditional, and then group both the conditionals into one statement. Finally, there is the not operator !, which negates a conditional clause (again, note the parentheses!):

```
if (!(wearingSeatbelt == "n")){
           cout << "The driver was wearing a seatbelt!\n";
```

```
}
```

Challenge 2:

Finish implementing the seat belt law for the driver who goes over 40 mph (just follow the format above). Then come up with another law and add it to your code. Be as creative and ridiculous as you want!

Source