5.4 — Goto statements

BY ALEX ON JUNE 21ST, 2007 | LAST MODIFIED BY ALEX ON JULY 2ND, 2015

The **goto statement** is a control flow statement that causes the CPU to jump to another spot in the code. This spot is identified through use of a **statement label**. The following is an example of a goto statement and statement label:

```
#include <iostream>
#include <cmath> // for sqrt() function

int main()
{
    double x;
tryAgain: // this is a statement label
    std::cout << "Enter a non-negative number";
    std::cin >> x;

    if (x < 0.0)
        goto tryAgain; // this is the goto statement

    std::cout << "The sqrt of " << x << " is " << sqrt(x) << std::endl;
    return 0;
}
```

In this program, the user is asked to enter a non-negative number. However, if a negative number is entered, the program utilizes a goto statement to jump back to the tryAgain label. The user is then asked again to enter a new number. In this way, we can continually ask the user for input until he or she enters something valid.

In the section on variables, we covered three kinds of scope: local (block) scope, file scope, and global scope. Statement labels utilize a fourth kind of scope: function scope. The goto statement and its corresponding statement label must appear in the same function.

There are some restrictions on the use of goto statements. For example, you can't jump forward over a variable that's initialized in the same block as the goto:

```
int main()
{
    goto skip; // invalid forward jump
    int x = 5;
skip:
    x += 3; // what would this even evaluate to?
    return 0;
}
```

In general, use of goto is shunned in C++ (and most other high level languages as well). **Edsger W. Dijkstra**, a noted computer scientist, laid out the case in a famous but difficult to read paper called **Go To Statement Considered Harmful**. The primary problem with goto is that it allows a programmer to cause the point of execution to jump around the code arbitrarily. This creates what is not-so-affectionately known as spaghetti code. **Spaghetti code** is code that has a path of execution that resembles a bowl of spaghetti (all tangled and twisted), making it extremely difficult to follow the logic of such code.

As Dijkstra says somewhat humorously, "the quality of programmers is a decreasing function of the density of go to statements in the programs they produce".

Goto statements are common in some older languages, such as Basic or Fortran, and even used in C. However, in C++, goto statements are almost never used, as almost any code written using a goto statement can be more clearly written using other constructs in C++, such as loops, exception handlers, or destructors (all of which we'll cover in future lessons).

Rule: Avoid use of goto statements unless necessary

5.5 -- While statements