Using ‘goto’ correctly

Summer 2018

The ‘goto’ keyword and labels

- The C language has a goto statement
  - Must be used carefully, considered bad style by most people
  - Can make programs hard to understand or debug if misused
  - Java does not have goto
- goto works like an unconditional branch in assembly language
- Lets you get out of multiple scope levels immediately
- The destination of goto is indicated using a label
  - A label is a name (identifier) ending with a colon ‘:’ located at the start of a line

Acceptable uses of ‘goto’

- There are only two situations in modern programming where using goto can be justified
  1. To “break” out of nested scopes (loop or switch)
  2. Error handling / recovery
- The 2nd use can be seen in Microsoft example code
- Avoid branching to a label that’s *earlier* in the function
- If you are in doubt, or you can easily avoid using goto, then don’t use goto!
- The compiler may complain if you use goto incorrectly

‘goto’ example

```
#include <stdio.h>
#include <stdbool.h>

int main(int argc, char** argv)
{
    // Input characters ("commands") until the user enters 'q'
    while (true) {
        char ch;
        scanf("%c", &ch);

        switch (ch) {
            case 'a':
                printf("Executing command 'a'\n");
                break;

            case 'b':
                printf("Executing command 'b'\n");
                break;

            case 'q':
                printf("Executing command 'quit'\n");
                goto quit;
                break;               // Note: This statement is never reached
        }
    }

quit:
    printf("Program terminating\n");
    return 0;
}
```

Label

Unconditional branch