If `v` doesn't equal 0, the statements clinging to the underside of the `while` loop are skipped, just as though you had written this:

```
if(v==0)
```

There's an exception, of course — a kind of loop you can fashion that always executes *at least once*. It's the upside-down `while` loop, called a `do-while` loop. This type of loop is rare, but it has the charming aspect of always wanting to go through with it once.

The devil made me `do-while` it!

The following program contains a simple `do-while` loop that counts backward. To add some drama, you supply the number it starts at:

```
/* An important program for NASA to properly launch
America's spacecraft. */

#include <stdio.h>

int main()
{
    int start;

    printf("Please enter the number to start\n");
    printf("the countdown (1 to 100):");
    scanf("%d",&start);

/* The countdown loop */

    do
    {
        printf("T-minus %d\n",start);
        start--;
    }
    while(start>0);

    printf("Zero!\nBlast off!\n");
    return(0);
}
```

Type this source code into your editor. Save the file to disk as COUNTDWN.C.

Compile. Fix any errors. Notice that a semicolon is required after the end of the `do-while` loop. If you forget it, you get an error.