The escape sequences used in string constants are also legal in character constants, so `'\n'` stands for the value of the newline character, which is 10 in ASCII. You should note carefully that `'\n'` is a single character, and in expressions is just an integer; on the other hand, `'\n'` is a string constant that happens to contain only one character. The topic of strings versus characters is discussed further in Chapter 2.

Exercise 1-8. Write a program to count blanks, tabs, and newlines.

Exercise 1-9. Write a program to copy its input to its output, replacing each string of one or more blanks by a single blank.

Exercise 1-10. Write a program to copy its input to its output, replacing each tab by `\t`, each backspace by `\b`, and each backslash by `\\`. This makes tabs and backspaces visible in an unambiguous way.

1.5.4 Word Counting

The fourth in our series of useful programs counts lines, words, and characters, with the loose definition that a word is any sequence of characters that does not contain a blank, tab or newline. This is a bare-bones version of the UNIX program `wc`.

```
#include <stdio.h>

#define IN   1  /* inside a word */
#define OUT  0  /* outside a word */

/* count lines, words, and characters in input */
main()
{
    int c, nl, nw, nc, state;

    state = OUT;
    nl = nw = nc = 0;
    while ((c = getchar()) != EOF) {
        ++nc;
        if (c == '\n')
            ++nl;
        if (c == ' ' || c == '\n' || c = '\t')
            state = OUT;
        else if (state == OUT) {
            state = IN;
            ++nw;
        }
    }
    printf("%d %d %d\n", nl, nw, nc);
}
```

Every time the program encounters the first character of a word, it counts one more word. The variable `state` records whether the program is currently in a word or not; initially it is ``not in a word'', which is assigned the value `OUT`. We prefer the symbolic constants `IN` and `OUT` to the literal values 1 and 0 because they make the program more readable. In a program as tiny as this, it makes little difference, but in larger programs, the increase in clarity is well worth the modest extra effort to write it this way from the beginning. You'll also find that it's easier to make extensive changes in programs where magic numbers appear only as symbolic constants.