Assignment 11: Reversing Words

11.1 Scanning in a String

You may scan a string for input data as if it is the standard input (`stdin`). The standard library function `sscanf` does this task of scanning. Accepting a char pointer as the first argument, `sscanf` does the same operation of `scanf` treating the string argument as input.

If several scanning operations should be performed on the same string buffer, it is better to use the format string `%n` for `sscanf`. The format string `%n` is to store the number of characters scanned up to that point in the current `sscanf` call. Therefore, the updated location in the current string buffer can be calculated using the value obtained with `%n`. The following code shows a typical usage of `sscanf`:

```c
#include <stdio.h>

int main()
{
    char line[] = "10 20 30 40", *buf = line;
    int sum = 0, n = 0;

    for (int pos = 0; sscanf(buf, "%d%n", &n, &pos) == 1; buf += pos) {
        printf("%3d %4d\n", n, sum += n);
    }

    return 0;
}
```

The starting location to scan in the `line` is pointed by `buf`. The pointer `buf` is updated using `pos`, the value of which is set by `sscanf` as specified with `%n`. There is a dual function of `sscanf` for output, namely `sprintf`. Since `sprintf` returns the number of characters written, no format specifiers like `%n` are required for `sprintf`.

11.2 Programming Assignment 11: revwords.c

Write a program reversing and printing the words in the input line. Though every word is printed in reverse, the order of the words should be preserved. An input word consists of alphabets, numbers, and punctuation symbols: `?`, `!`, `.`, `,`, `;`, `:`, `'`, `"`, `-`, and `_`. Assume that a word contains at least one alphanumeric character.

Your program should eliminate punctuation symbols from the words. For this purpose, define and use the function `filtersym` with the following prototype:

```
String filtersym(String word);
```

where `String` is the type for character strings defined in the previous lab. Given the argument `word` representing `"can't!"`, for example, the function `filtersym` should return the string representing `"cant"`.

Your program is to read from standard input. The input consists of a single line containing several words separated by a space. The number of words is greater than zero. Assume that the maximum length of the input line is 256 including the newline characters. Your program should print to standard output. The output consists of a single line containing the reversed words with punctuation symbols eliminated. The reversed words should be printed in the order of the input line, and separated by a space.

Additional requirements for bonus points

- Use the standard library function `fgets` to read the input line.

- Define and use a function counting the number of punctuation symbols in a string.

Input	**Output**
`Don't stop now!!`	`tnoD pots won`
`You know C programming!`	`uoY wonk C gnimmargorp`