CS270 - System Software
Assignment #2
Fall 2016

The purpose of this assignment is to write a function that will *parse* a string into *tokens* (or *words*), similar to what the shell does.

For this assignment, you are to write a function with the following prototype:

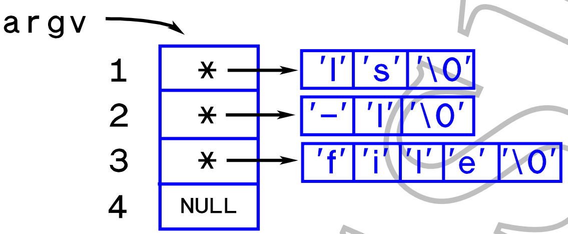
```
int makearg(char *s, char ***args); // or int makearg(char s[], char **args[]);
```

(Yes, this *is* the correct prototype - don't change it!!!) This function should accept a (c-style) string and a pointer to a pointer to `char` (or, if you prefer, a pointer to an array of pointers to char) (i.e., a pointer to the same type as `argv` in a C program), and should return the pointer defined to point to an array pointing to the separate tokens on the command line, as well as the number of tokens found as the function return. If some problem occured during the operation of the function, the value returned should be `-1`.

For example, if you were to call the function using the following code:

```
char **argv, str[] = "ls -l file";
int argc;
argc = makearg(str, &argv);
```

it should produce the following situation in memory:



The important part of this exercise is the function you write. However, for consistency in grading, your main program should be written to input a line, call the function, print out the number of arguments found, and then finally print out the arguments, one per line. The printing must occur within the main program (*not* in your subroutine), to prove that the parsed string has indeed been properly passed back to main.

Note that, as in the example above, the value of the pointer argument is undefined when it is passed to the `makearg` function. `makearg` should allocate any necessary dynamic memory necessary for use by the program. (The array used for the unparsed string need not be dynamically allocated.)

You should submit the program using `cscheckin`.