COSC 301: Operating Systems - Lab 3

For this lab, there's just one C function to write: `tokenify`.

`tokenify` should take a C string and split it up into whitespace-delimited words. The words should be returned as an array of C strings, with the last element of the array explicitly set to NULL. For this problem, you will need to allocate new chunks of memory using `malloc`. *The array, and each C string referred to in the array should be newly allocated from the heap.* You can use the `strdup` C library call, or any other C library call, if you wish. You should *not* modify the string passed into the `tokenify` function.

For example, if you get the string `go red sox` on input, you should return an array of *four* elements. The first array element should be a pointer to the string `go`, the second array element should point to the string `red`, the third array element should point to the string `sox`, and the fourth element should be NULL. Note that the memory for each of the non-NULL strings should be newly allocated from the heap. Since there's no built-in way to detect the length of an array, we are using the last element of the array of strings as NULL to indicate the end of the list/array.

You can either do the tokenization yourself (*i.e.*, find each whitespace delimited word using primitive comparisons), or you can use the C library function `strsep` (or `strtok`, or the thread-safe `strtok_r`). I'd recommend using one of the library functions: `strsep` will do the "hard" work of finding each space-delimited word, leaving you with the task of putting them into an array. The downside to the convenience of using `strsep` is that it can be a bit tricky; you'll want to carefully read the `man` page.