# Tokenizing Review

1. Firstly the code begins by including necessary header files:

stdio.h: for standard input and output functions.

string.h: for string-related functions.

1. It defines a macro BUFFER\_SIZE with a value of 300.
2. It includes the header file tokenizing.h, which contains the function declaration for tokenizing.
3. The function tokenizing is defined:

It prints a message then it declares an array of characters named words with a size of BUFFER\_SIZE, which will store the user's input then it declares a character pointer nextWord and initializes it to NULL. This pointer will be used to traverse through the tokenized words.

It also declares an integer variable wordsCounter, which will keep track of the word number during tokenization.

1. Then the code enters a do-while loop that continues until the user enters "q" as the input.
2. Within the loop, it asks the user to input a few words separated by spaces.
3. It reads the user's input using fgets and stores it in the words array. The BUFFER\_SIZE ensures that the input does not exceed the size of the array.
4. It removes the trailing newline character from the input by replacing it with the null terminator '\0'. This ensures that the input is properly formatted.
5. It also checks if the user's input is not equal to "q" so that it means that the user wants to continue, not quit.
6. If the user didn't input "q", it proceeds to tokenize the input, it uses strtok to tokenize the words array using space (' ') as the delimiter then strtok returns a pointer to the next token (word) in the string until there are no more tokens, where it returns NULL. Then it enters a while loop that iterates through the tokens and prints each word along with its word number. The word number is stored in the wordsCounter variable and increments after each word is printed.
7. After printing all the tokens from the input line, it goes back to step 6 to ask for the user's input again unless the user enters "q" to quit.
8. Once the user enters "q", the loop terminates, and the code prints a message indicating the end.

Here are the library functions used:

1. printf
2. fgets
3. strlen
4. strcmp
5. strtok