3. C programming: Word Count (30 min)

Your task is to compute the frequency of each English word occurring in an input string. There could be one or more white spaces to separate words in the string. A white space is any character **c** for which **isspace(c)** returns **true**. White spaces can also occur at the beginning or the end of the string. For instance, the string **“ the Systems Programming course ”** contains the following words: “**the**”, “**Systems**”, “**Programming**”, and “**course**”. Each (*word*, *frequency*) pair will be stored in a single linked list whose structure is given as follows.

```
struct WordCount {
        char *word;                 /* an English word */
        int count;                  /* the frequency of the word */
        struct WordCount *next;     /* pointer to next element in list */
}
```

You need to write three functions with the following signatures.

(a) ***struct*** *WordCount **create_node**(**char ***word, int count**)*;

This function returns a node with a given *word* and its frequency *count*. The *word*’s content must not be copied into a new memory i.e., simply copying the pointer is sufficient. In case of invalid parameters, the function returns ***NULL.***

(a) ***void*** *convert_to_lowercase* ***(char*** *word**)***;

This function converts an input word to lowercase.

(c) ***struct*** *WordCount **count_words**(**char ***s**)*;

This function takes a string **s** as input and returns a list of English words occurring in the string together with their corresponding frequencies. The word comparison is **not case-sensitive**. All the words in the list must be lower-case. The order of words in the resulting list must obey that in the input string. The function returns **NULL** if the input string is **NULL** or empty. You can assume that if the string is not NULL and not empty then it only contains valid English words and white spaces used to separate the words. For example, with the input **“ hello World heLLo wOrld ”** the function should return a word-count list whose content is given as below.

All **functions** have to **check that** their **parameter values** are **valid**. Make sure that **all allocated memory is freed** correctly, especially when an error occurs. Check the return values for function calls where errors can occur and handle them appropriately. You can use any function of the standard C library. Some functions you may find useful are listed below.

int tolower **(int** c**):** If c is an upper-case letter, **tolower** returns the corresponding lower-case letter. If **c** is not an upper-case letter, **c** is returned unchanged.
int *strcmp **(const char** *s1**, const char** *s2**):** The **strcmp** function compares the string **s1** against **s2**, returning a value that has the same sign as the difference between the first differing pair of characters (interpreted as unsigned char objects, then promoted to int). If the two strings are equal, **strcmp** returns 0**.**