Addendum to Q1

An alternative to Q1 that makes writing Q3 easier (either approach is acceptable)

Write the following functions:

- A function with the signature `int letter_count(char *)`
    - that counts the number of letters in the string (characters between A/a to Z/z)

- A function with the signature int * create_freq_table()
    - that return an empty frequency table
        - i.e. a 1d array of 26 elements
    - each element should be initialized to zero

- A function with the signature void add_letters(int * freq_table, char * string)
    - each element of the array holds the numbers of times each letter (upper or lower case) occurs in the string that is passed in as an argument
    - i.e. the first index holds the count for 'a' / 'A', the second index hold the count for 'b' / 'B', etc.
    - The string is looked at character by character, and if the character is a letter, the appropriate count is updated

If you implement these functions, the function `frequency_table(char * string)`need not be written.

Addendum to Q2

In Question 2, the following is stated:
"Let ENGLISH_FREQ[*i*] (which you can also denote EF[i])
be an array that stores the above table, where *i* = 0 to 25.
…
Note: the table should be a constant throughout your program."

This implies that EF should be an array created using a #define macro and placed in a .h file. However, you cannot create an array using a #define.

There are two approaches to handling this ambiguity. You can create a global variable

```
int EF[26] = {0.08167, 1.492, 2.782, …};
```

which, while not created by a #define, can be treated as a constant and so given an uppercase name; or you can use

```
#define EF {0.08167, 1.492, 2.782, …}
```

and then implement a local array

```
int ef[26] = EF;
```

which can be passed into any function that needs it.

Either approach is fine.