MATLAB Programming NCTU Fall 2017 LAB#7 2017/11/13

For this lab, you will modify the work done in the previous lab to get a little experience on MATLAB structures. Some parts in your function are the same as the previous lab, including:

- Read the text file (use the same text file as the last lab).
- Extract the individual words.
- Identify the unique words and their counts.

Below explains what are different from the previous lab:

The function header becomes

```
A = mylab8(fn, sort_mode)
```

Here `fn` is the path to the file. The output `A` is an array of structures with three fields: `word` is the word itself (an array of characters), `len` is the length of the word, and `count` is its times of occurrence. Example:

`A(1).word` is `'she'`, **`A(1).len`** is 3, and **`A(1).count`** is `4`.

`A(2).word` is `'sells'`, **`A(2).len`** is 5, and **`A(2).count`** is `4`.

The second input **`sort_mode`** is a string that indicates how the words should be sorted. It can be `'none'`, meaning no sorting; `'word'`, meaning sorting in dictionary order; `'len'`, meaning sorting by length; `'count'`, meaning sorting by count.

You can use **`[A.field]`** to put all the values of the same field in the structure array into a vector, and sort this vector. This applies to **`[A.len]`** and **`[A.count]`**. For sorting the words, use **`{A.word}`** to put all the words into a cell array of strings, and they apply **`sort`** to this cell array.

The following is an example illustrating how to use sorted indices to reorder the array elements:

```
[~, k] = sort([A.len]);
A = A(k);
```

Finally, if you call the function without an output argument, let the function print out the words and their counts (in the sorted order if applicable).