A Zero Marks Quiz on Dictionaries

Question 1

(10 fake marks) Write a function called `read_words(fname)` that reads a file of words, one per line, and returns them in a dictionary. The keys are the words, and the values are all 1. There could be duplicate words in the file. For word appears more than once, then ignore the repeats.

Make sure to strip-off any whitespace characters from the beginning and end of each word.

For example, suppose `words.txt` contained these words:

```
shoes
hats
glove
shoes
socks
```

Then:

```
>>> word_dict = read_words('words.txt')
>>> word_dict
{'shoes': 1, 'hats': 1, 'glove': 1, 'socks': 1}
```

Your answer should use correct syntax, correct and consistent indentation, and general good Python style. Your code should **not** do any unnecessary work.

Sample Solution

```
def read_words(fname):
    all_words = {}
    word_file = open(fname)
    for w in word_file:
        w = w.strip()
        all_words[w] = 1
    return all_words
```

Marking Scheme

- **1 mark**: correct function header
- **1 marks**: initializing the dictionary
- **1 mark**: opening the file
- **1 marks**: getting each word from the file using a loop
- **1 mark**: stripping whitespace from the word
- **2 marks**: correctly adding the word to the dictionary with value 1
- **1 mark**: correct result returned
- **2 marks**: overall good indentation, syntax, and style

-1 mark for any unnecessary code.

Question 2

(10 fake marks) Write a function called `get_misspelled(line, word_dict)` that returns an *alphabetically sorted list* of all the misspelled words in the string `line`. A word is considered misspelled if it is *more than one character long* and also *doesn't appear as a key* in `word_dict`.

If the same misspelled word appears more than once, then only include it once (so there are no duplicates in the returned list).

For example, assuming `words.txt` is the same as in the previous question:

```
>>> words = read_words('words.txt')
>>> get_misspelled('I like socks and I like gloves', words)
['and', 'gloves', 'like']
```

Your answer should use correct syntax, correct and consistent indentation, and general good Python style. Your code should **not** do any unnecessary work.

Sample Solution

```
def get_misspelled(line, word_dict):
    misspelled = {}
    for w in line.split():
        if len(w) > 1 and w not in word_dict:
            misspelled[w] = 1
    unique_words = list(misspelled.keys())
    unique_words.sort()
    return unique_words
```

Marking Scheme

- **1 mark**: correct function header
- **1 mark**: initializing the result
- **2 marks**: correctly looping through the words in the line
- **2 marks**: recognizing if a word is misspelled and adding it to the result
- **2 marks**: returning the correct sorted list
- **2 marks**: overall good indentation, syntax, and style

-1 mark for any unnecessary code.