# CSCI2100B: Data Structures (Spring 2011)

Assigned: *01 Mar 2011* Due: *18 Mar 2011 23:59*

## Programming Assignment 2: Misspelling

Limits: Runtime - 2 secs • Memory - 64 MB • Submission - 15 times

### Background

Scanning for misspelled words from a long article is a tedious task for us. Usually embedded with word processing software, a spell checker is implemented to efficiently verify the correctness of every word in the article against a given dictionary.

Since there are many words in the article and the dictionary, the design of the spell checker is very challenging. In this assignment, the largest dictionary may contain more than 100,000 words and the input text can be a long novel like "the Mysterious Cases of Sherlock Holmes", that contains more than 1,100,000 English words. To finish the spell check in a second requires a careful design of the algorithm with proper data structures.

### Problem

Quickly identify *ALL* misspelled words (those absent in the dictionary) in the given passage and display the list of misspelled words in their order of first appearance.

### Input

The input is divided into two parts: the dictionary and the passage.

The dictionary is a set of valid words in either lower or upper cases. Each word is written in a separate line, terminated by a newline character. The end of the dictionary is specified by a single '#' (which is not a part of the dictionary). There will be at most 150,000 words in the dictionary.

The input is then followed by a passage containing only ASCII characters. A line in this part can be arbitrarily long and is terminated by a newline character. Punctuation (`,.!?`) and whitespaces like spaces and tabs can appear in the passage but should be ignored during spell check.

A word is considered to be consecutive ASCII English characters (`a - z`, `A - Z`). The maximum number of characters in a word is **50**. The spell checking should be case-insenitive (i.e. `apple` and `AppLe` are considered the same).

Sample Input:

```
I
am
a
boy
#
I␣am␣a␣good␣boy.
"I␣am␣an␣boyy."
I␣an␣a␣'goo'd␣boy.
```

## Output

Print each misspelled word in lower case on a separate line, following their order of first occurrence in the passage. Ignore and skip repeated typos.

Sample output:

```
good
an
boyy
goo
d
```

## Test Cases & Scoring

Your program will be tested against a set of 5 test cases, namely:
1. The sample input
2. Small dictionary (< 100 words) and short passage (< 200 words)
3. Normal dictionary (< 5,000 words) and normal passage (< 30,000 words)
4. Large dictionary (< 150,000 words) and large passage (< 1,000,000 words)
5. Large dictionary (< 150,000 words) and huge passage (> 1,000,000 words)

Each test case scores 2 mark. Your program can score 10 marks in total.

## Submission

You should write your program in a single C source file. Submit your program using your UNIX account, following the instructions below:

1. Copy/upload it to your CSE UNIX account.
2. SSH to any CSE UNIX workstation through SSH, compress the source code by
   `gtar zcvf <sid>.tar.gz <file_name>.c`
   where `<sid>` is your 10-digit student ID and `<file_name>` is the name of the source code you wish to submit. Do **NOT** include other files.
3. Submit it to our Judge System by
   `uuencode <sid>.tar.gz <sid>.tar.gz | mailx -s "ASG<code> <sid>" csci2100b`
   where `<sid>` is your 10-digit student ID and `<code>` is the assignment code you are trying to solve.
4. Upon successful submission, you will receive a submission receipt and the judge reply very soon. You should **KEEP** your submission receipt for future references.

- End -