# COMP 597/CMPSC 497 – Natural Language Processing
# Homework #2
## Fall 2013

**Due Date:** Wednesday, October 9, 2013
**Total Points:** 100

In this assignment you are going to implement in Perl a spelling correction application for non-word spelling errors. Given a text file as an input, your program should:

- Detect the non-words
- Generate the three most likely correct candidates for each non-word
- For each non-word in the text, display the three candidates and how they are aligned with the misspelled word.
- The output of your spell checker should be a text file where the information displayed (the misspelled word, the three likely candidates and their three alignments) should be organized and clearly shown.

## Spelling checker:

You are going to use the minimum edit distance function to implement your spelling checker. The algorithm should return the minimum alignment cost and how the two strings are aligned together. You are going to have two versions for this algorithm:

- **Method A:** uses 2 as a cost for substitution and 1 as a cost for either deletion and insertion.
- **Method B:** in addition to the insertion, deletion and substitution costs, you are going to use a nearby-keys matrix of costs that you should come up with.

## Dictionary:

In the assignment folder you will find two text files that you should use as your spelling checker dictionary.

1) **shakespeare.txt**: The complete works of Shakespeare tokenized so that there is a space between words and punctuation.
   In order to work with this dictionary, you need to:
   - preprocess it by having each word stored on a separate line
   - remove some of the punctuations that are not needed for the spelling correction task. Make sure to keep the useful punctuations, such as **'** and **-**.
2) **wordlist.txt**: A word list used by scrabble and word game players.

## Evaluation:

You are going to evaluate both of your spelling correction implementations (method A & Method B) on the two dictionaries described above. To properly evaluate each implementation, you should:

1. Come up with text files that will check the correctness of your spell checker. Each file should have a text that includes some spelling errors. To simplify your program, use text that is tokenized in a way similar to the shakespeare.txt file. Your text examples should not be taken from shakespeare.txt. The following link has a list of common English spelling errors: Peter Norvig's list of errors. Make sure to only use the non-word errors.
2. Run your program on each text file then document the following:
    a. The number of spelling errors that were detected in the text
    b. The total number of spelling errors found in the text
    c. The percentage of error detection
    d. Compare the likely candidates produced from method A to the ones produced by method B. Justify which method was more accurate.

Apply this evaluation process first using the Shakespeare dictionary then using the word list dictionary. Make sure to explain in your evaluation report which dictionary gave better results.

## What to hand in:

- The two Perl implementations of the spelling checker. (50 points)
- The text files you used to evaluate your programs. (15 points)
- Your report (either PDF or MS word format) that describes the evaluation results. (25 points)
- The nearby-keys matrix of costs. Please explain why you chose these costs. (10 points)