# Homework Assignment 2

## LING 1330 Introduction to Computational Linguistics

The goal of this homework assignment is to write a **Past Tense Generator** script. We will do this in three parts.

### Part A.

Let's start by writing a script that handles the overall control flow of the program. Your job is to write a Python script that:

1. prompts for verbs, with a hint for how to exit the program
2. prints out **each verb** in the user input followed by **a tab** and then the **verb suffixed with 'ed'**
3. continues to do 1 and 2, until the user types in 'EXIT'
4. finally prints out a 'goodbye' message and then quits.

When run, your program should produce the exact same output as below:

```
>>> ================================ RESTART ====================
>>>
What are your verbs? (Type EXIT to quit): walk learn rest fill
walk  walked
learn learned
rest  rested
fill  filled
What are your verbs? (Type EXIT to quit): eat pray love
eat   eated
pray  prayed
love  loveed
What are your verbs? (Type EXIT to quit): understand
understand   understanded
What are your verbs? (Type EXIT to quit): EXIT
Thank you for using the Past Tense Generator. Goodbye.
>>>
```

For now, do not worry about getting the correct past tense form: simply output the base form with *-ed* suffixed at the end. You will of course get incorrect forms such as **eated* and **loveed*, which we will take care of in Part B.

Getting the overall control flow of the program is the point of Part A. When your script works as it should, save a copy, and then move on to Part B.

## Part B.

There are, in fact, many rules involved in generating the correct past tense form of a given verb. Your job is to extend your Python script so that it handles the following cases:

| | BASE FORM | PAST TENSE | Note |
|---|---|---|---|
| a. | *walk*<br>*learn*<br>*pray* | *walked*<br>*learned*<br>*prayed* | Standard case: no special handling necessary beyond the *-ed* suffixation |
| b. | *go*<br>*buy*<br>*break*<br>*sit* | *went*<br>*bought*<br>*broke*<br>*sat* | The past tense form is unique to each lexical item. Make sure your script correctly handles at least these 15: *come*, *eat*, *sleep*, *see*, *pay*, *sing*, *tell*, *get*, *teach*, *feel*, *hear*, plus the 4 shown on the left. |
| c. | *hit*<br>*cost*<br>*spread* | *hit*<br>*cost*<br>*spread* | The past tense form is the same as the base form. Make sure your script handles at least these 10: *cut*, *put*, *let*, *hurt*, *quit*, *read*, *broadcast*, plus the 3 shown. |
| d. | *live*<br>*celebrate* | *lived*<br>*celebrated* | Base form ends in *e*: attach *-d*. |
| e. | *dry*<br>*apply* | *dried*<br>*applied* | Base form ends in a "consonant" character followed by *-y*: change *y* to *i* before suffixing *-ed*. |
| f. | *tap*<br>*plan*<br>*shred*<br>*swat* | *tapped*<br>*planned*<br>*shredded*<br>*swatted* | A monosyllabic word ending with a "short vowel" followed by a single "consonant": double up the final consonant.<br><br>Make sure the rule does not apply to the ones in the bottom row.<br>(There's more to this consonant duplication rule. See the footnote[1] below.) |
| | *beam*<br>*flaw*<br>*fix* | *beamed*<br>*flawed*<br>*fixed* | |

Built to the specifications above, your Past Tense Generator should be pretty competent. But it is far from complete. The following errors, which are beyond the present scope, are allowed:

- Any irregular verb not included in the 15 in b.:
  *understand* → **understanded* instead of *understood*
- Any irregular verb not included in the 10 in c.:
  *set* → **setted* instead of *set*
- Any multi-syllabic word that ends with a stressed short vowel and a single consonant:
  *defer* → **defered* instead of *deferred*

You may implement cases a. – e. without moving on to Part C. For the f. case, which is the most complex of all, refer to Part C.

[1] Duplication of the final consonant only happens in a stressed syllable. In multi-syllabic words, therefore, the process applies only to the words with the primary stress falling on the very last syllable: *committed*, *occurred*, *controlled*. Compare them with *happened*, *traveled* and *remembered*, where the final syllable is unstressed. There are exceptions to this pattern, unfortunately: *worshipped*, *kidnapped*.

## Part C.

In implementing the f. case, build and use your own function named `isLikeTap()`. This function takes a verb base form as a string and returns True/False. It returns:

- **True** if the given verb is like *tap*, that is, (1) the verb is monosyllabic and (2) it ends with a short vowel and a single consonant.
- **False** otherwise.

The function should work as follows:

```
>>> isLikeTap('tap')
True
>>> isLikeTap('stop')
True
>>> isLikeTap('shred')
True
>>> isLikeTap('swat')
True
>>> isLikeTap('beam')
False
>>> isLikeTap('flaw')
False
>>> isLikeTap('fix')
False
>>> isLikeTap('pray')
False
>>> isLikeTap('ab')
False
```

Using this function for case f. should be straightforward. Basically, if the value of `isLikeTap(word)` is True for your `word` variable, then you want to double up the final consonant and then attach the *-ed* suffix.

Make sure to **test** your script thoroughly. Also, provide proper documentation on your code by inserting **comments** where appropriate. When you are done, upload your Python script.

## 5 BONUS POINTS

Implement the entire "past tense building" part of the code as a function named `getPastTense()`. This function takes a single verb and then returns its correct (sometimes incorrect) past tense form as a string:

```
>>> getPastTense('walk')
'walked'
>>> getPastTense('buy')
'bought'
>>> getPastTense('hit')
'hit'
>>> getPastTense('live')
'lived'
>>> getPastTense('apply')
'applied'
>>> getPastTense('plan')
'planned'
>>> getPastTense('beam')
'beamed'
>>> getPastTense('fix')
'fixed'
>>> getPastTense('understand')
'understanded'
>>> getPastTense('defer')
'defered'
```