# Lab 4: Dictionaries (To be submitted before 4th July 2020)

## The Problem, really Puzzle

Here is a puzzle. You are given a word and your task is to find the anagram to that word. In case you don’t remember, an anagram is the rearrangement of a word’s letters to make a new word. For example, ‘parental’, ‘prenatal’, and ‘paternal’ are all anagrams of one another

## Your Tasks

Here are 14 words. Find an anagram of those words starting with the letter ‘v’. Included in the directory is a wordList file of about 230,000 English words. You can use it to look up anagrams.

serve, rival, lovely, caveat, devote, irving, livery, selves, latvian, saviour, observe, octavian, dovetail, levantine

## Hints, doing it in steps

1. We are going to use Python’s dictionary data structure. We will store a key-value pair in the dictionary that we can use to solve our problem. Consider two anagrams we know. The words are ‘serve’ and ‘verse’. Let’s create a string of the sorted list of the letters in each word. Thus ‘serve’ is transformed to ‘eersv’, the sorted list of its letters. Now think!!! What would ‘verse’ become under the same transformation? How can I use that fact to find two anagrams in a list of words? The **sorted letters are our ke**y in the dictionary
2. How do we do this conversion? We cannot sort a string, but we can sort a list. So we need to:
   1. convert a string to a list myList = list('serve')
   2. sort the list myList.sort()
   3. turn the list back into a string sortedString=''.join(myList)
      1. the join method takes the elements of a list and attempts to put them together into a string with the string before the dot placed in-between each list element. So ':’.join(['a','b','c']) returns 'a:b:c'. But joining with an empty string just yields the joined string. Thus ''.join(aList) is another idiom: one for making a string from a list of strings. (Try it: type ''.join(['a','b','c']) in the Python shell.)
3. Now, open up the file called ‘wordList.txt’, which is in the lab directory, and examine the words. Store each word that starts with `v’ in a dictionary using the sorted letters in the word as key. To open the file, we use a for statement such as:
   1. fp = open('wordList.txt')

for word in fp:

suite

This command will set the variable word to each line in the file wordList.txt, which as it turns out is one word. Note that you can ignore any word that does not being with the letter ‘v’

1. Now your job is to take each of the 14 words and, one at a time, find the anagram of the word you are considering. Use the facts from steps one and two to make that decision.
2. Repeat this for all the words we are working with (the full list of 14) and discover the anagram of that word that begins with ‘v’.

One last thing to consider: If you take a list of tuples and sort them, how would they be sorted? Try

myList = [(2,300),(17,400),(1,600),(4,1100)]

myList.sort()

print(myList)  
  
 myList = [('c',5),('a',9),('a',2),('b',8),('c',8), ('c',)]

myList.sort()

print(myList)

Try some other examples.

You need to be able to explain how a list of tuples is sorted in general.

**\*NOTE:** *LATE SUBMISSION WILL NOT BE ACCPTED*