

CSC401- Assignment 5

Jay Tang

Due Wednesday, May 18, 11:59pm

Reading

Read **Chapter 6** in Introduction to Computing using Python: An Application Development Focus, Second Edition by Ljubomir Perković.

Logistics

You need to do this assignment on a computer which has Python 3 installed on it.

[Python 3.10 download page can be found here.](#)

You are encouraged to work with your classmates on the assignments. If you do work with someone on the assignments, please include the name of your collaborators at the top of the file you submit. If you worked alone, please indicate that at the top of your submission. **A submission without collaboration information will not receive credit.**

A submission that includes code which does not run will not get any points for the part unless specifically documented reason of the error.

Submission

Submit the assignment using Assignment 5 folder. Submit only a **single python file** using your name as file name (e.g. Jay_Tang_Assign_5.py).

This assignment is due Wednesday, May 18, 11:59pm. Submissions after the deadline will be automatically rejected by the system.

Assignment

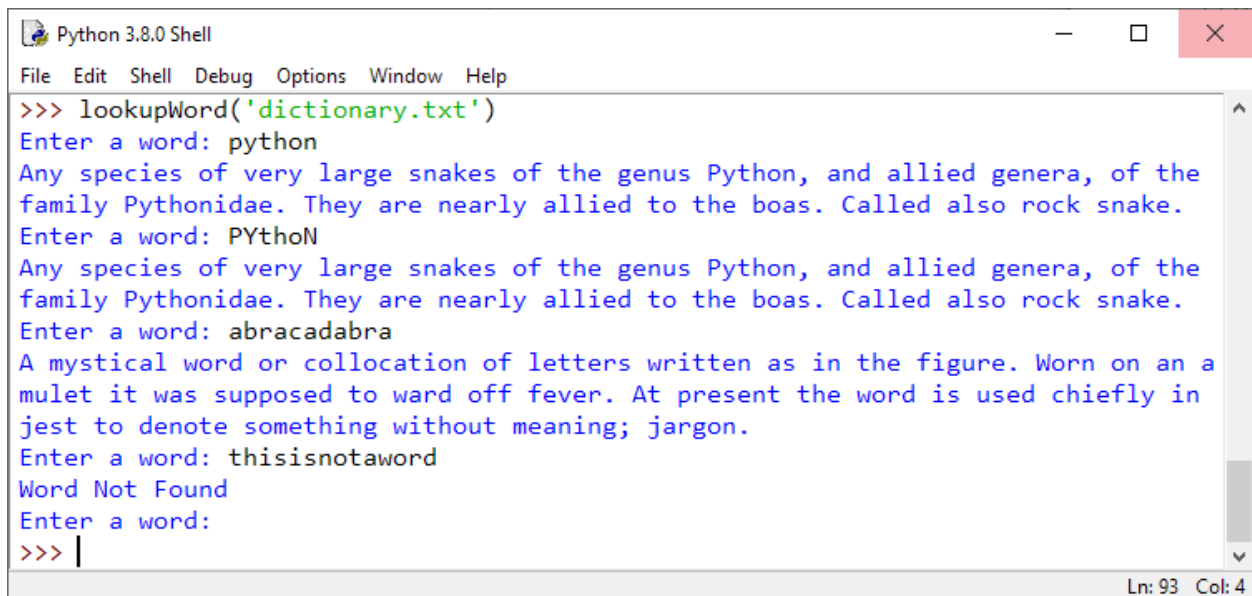
1. (70pt) Dictionary

a. (35pt) English dictionary

Write a function `lookupWord(filename)` that takes as a parameter a string representing the file name. Each line of the file contains a word followed by three colons (`: : :`) followed by the word's definition. Assume the file won't contain multiple entries for the same word. The function reads the words and associated definitions from the file and repeatedly allows the user to enter a word, printing the corresponding definition in response. If the word is not in the file, the function prints Word Not Found in response to that query. The function stops when the user presses enter without typing any text.

I suggest make the word lower case before adding it to the dictionary as a key. You don't want your dictionary to be case sensitive.

You can find an example `dictionary.txt` file in the submission folder.



```
Python 3.8.0 Shell
File Edit Shell Debug Options Window Help
>>> lookupWord('dictionary.txt')
Enter a word: python
Any species of very large snakes of the genus Python, and allied genera, of the
family Pythonidae. They are nearly allied to the boas. Called also rock snake.
Enter a word: PYthoN
Any species of very large snakes of the genus Python, and allied genera, of the
family Pythonidae. They are nearly allied to the boas. Called also rock snake.
Enter a word: abracadabra
A mystical word or collocation of letters written as in the figure. Worn on an a
mulet it was supposed to ward off fever. At present the word is used chiefly in
jest to denote something without meaning; jargon.
Enter a word: thisisnotaword
Word Not Found
Enter a word:
>>> |
```

Ln: 93 Col: 4

b. (35pt) Index

At the end of this and other textbooks, there usually is an index that lists the pages where a certain word appears. In this problem, you will create an index for a text but, instead of page number, you will use the line numbers. Implement function `index(filename, words)` that takes as input the name of a text file and a list of words. For every word in the list, your function will find the lines in the text file where the word occurs and print the corresponding line numbers (where the numbering starts at 1). You should open and read the file only once.

```
Python 3.8.0 Shell
File Edit Shell Debug Options Window Help
>>> index('raven.txt', ['raven', 'mortal', 'dying', 'evil', 'happy', 'craven'])
raven      44,53,55,64,78,97,104,111,118,120
mortal     30
dying      9
evil       99,106
happy
craven     52
>>> index('python.txt', ['python', 'program', 'system'])
python     1,2,3,4
program    3
system     1
>>> index('test.txt', ['i', 'dont', 'care'])
Invalid file name test.txt
>>>
```

You can find `python.txt` and `raven.txt` in the submission folder. Note that punctuations in `python.txt` is limited to only comma and period, while in `raven.txt` there are more punctuations. Remember to remove those punctuations when parsing for the words. Your output format does not need to match exactly as shown in the example. But the indexes should all be correct.

2. (30pt) Set

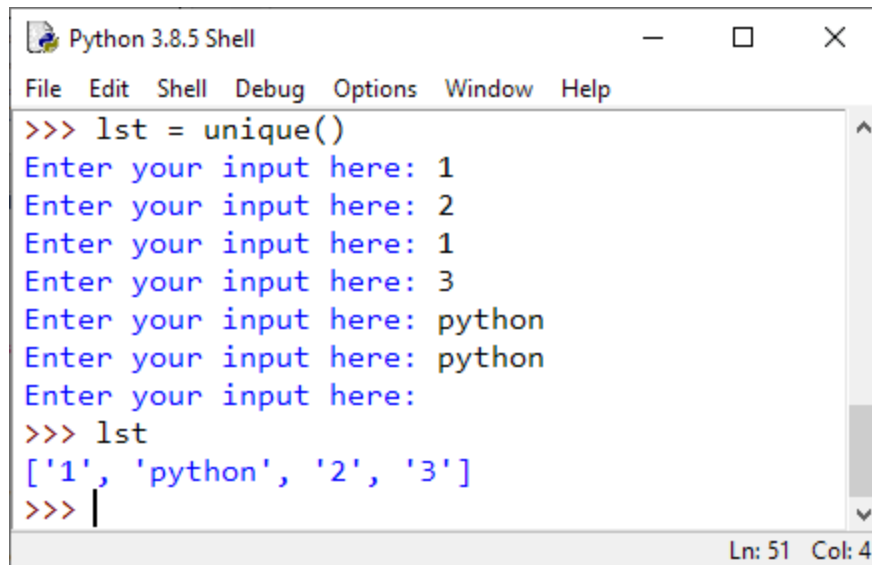
a. (15pt) Duplicates

Write a function `hasDuplicates(lst)` that takes a list as a parameter and returns `True` if there are duplicates in the list and `False` otherwise.

```
Python 3.8.5 Shell
File Edit Shell Debug Options Window Help
>>> hasDuplicates([1,2,3,4,5])
False
>>> hasDuplicates([1,2,3,2,4,3,5])
True
>>> hasDuplicates(['twinkle', 'twinkle', 'little', 'star'])
True
>>> |
```

b. (15pt) Unique

Write a function `unique()` that will keep asking user for inputs and return a list of unique inputs from the user. Use can end the input by hitting enter.

A screenshot of a Python 3.8.5 Shell window. The window has a title bar with the text "Python 3.8.5 Shell" and standard window controls (minimize, maximize, close). Below the title bar is a menu bar with the following items: File, Edit, Shell, Debug, Options, Window, and Help. The main area of the window contains a Python REPL session. The prompt is ">>>". The first line of code is "lst = unique()". This is followed by six lines of input prompts: "Enter your input here:". The inputs are "1", "2", "1", "3", "python", and "python". The next line is another prompt ">>>". The output is "['1', 'python', '2', '3']". The final line is a prompt ">>>" followed by a vertical bar cursor. On the right side of the main area is a vertical scrollbar. At the bottom right of the window, the status bar shows "Ln: 51 Col: 4".

```
Python 3.8.5 Shell
File Edit Shell Debug Options Window Help
>>> lst = unique()
Enter your input here: 1
Enter your input here: 2
Enter your input here: 1
Enter your input here: 3
Enter your input here: python
Enter your input here: python
Enter your input here:
>>> lst
['1', 'python', '2', '3']
>>> |
Ln: 51 Col: 4
```