# CSCI 331: Object Oriented Software Development

Lab #3 (100 points)

Due: Wednesday, February 22 by 23:59

**Directions: The following lab will help you explore how to write Python programs using object-oriented software development concepts. Please label your programs `q<num>.py`, where `num` is the question number. For example, your solution to the first question, will be stored in the file `q1.py`.**

**Your answers should be based on the content and procedures covered in class, and not on other implementations or descriptions that can be found online.**

**Your programs should be driven by logic. You will not receive credit if you use statements to jump out of a loop: break, continue, etc.**

**Take your time and make sure you understand everything in this document before getting started. Make sure your programs match the output EXACTLY as given for each question. This is important, as one of the keys to being a good programmer is attention to details. Finally, don't leave things to the last minute.**

## 1 Questions

1. Write a program (called `q1.py`) that creates a concordance from text files, and extends a Python class. A concordance is a list of the words present in a text, along with the line numbers in which they are found. More specifically, you will be parsing text files from Project Gutenberg.

   You will be extending Python's built-in dictionary **dict** class for this task. More specifically, you will write the code for the methods in the `Lab3` class:

   - **__setitem__()**: If a key **k** does not have a value **v** associated with it, associate an empty list first. Otherwise, append the value **v** at the end of this list. Keep in mind that this changes the default behaviour of the **dict** class.
   - **clear()**: Remove all items from the dictionary, and print the following message: `*+* Erasing data...  *+*`
   - **topKeys()**: takes an integer **n** as a parameter (with a default value of 100), and returns the keys where its values have a length greater than **n**.

   For example, this is a sample output from loading the **Lab3** to the interactive interpreter:

```
>>> test = Lab3()
>>> test['a'] = 0
>>> test
{'a': [0]}
>>> test['a'] = 1
>>> test['a'] = 5
>>> test
{'a': [0, 1, 5]}
>>> test['b'] = 44
>>> test
{'a': [0, 1, 5], 'b': [44]}
>>> test.topKeys(2)
['a']
>>> test.clear()
*+* Erasing data... *+*
```

**Example #1.** At the prompt, the user provides `txts/345.txt` as a command line argument (line 1). Random entries in the concordance are reported on lines 3-12. Top keys are printed on lines 14-33. The object is cleared and the output message is produced on line 34.

```
 1  python q1.py txts/345.txt
 2  ***** Sample entries: *****
 3  sprout [4822, 4822]
 4  Customs [3313, 3318, 9258]
 5  unholy [7790, 8624, 8788]
 6  problem [2801, 11079, 14306, 15739]
 7  stricken [12585, 12604, 13554, 14916]
 8  sophistic [9534]
 9  thocht [14177, 14209]
10  _continued_ [608, 1110, 1609, 6581, 7883, 8476, 8914]
11  athwart [3077]
12  vivify [13012]
13  ***** Top Keys: *****
14  Jonathan 191
15  Harker 158
16  Seward 126
17  Helsing 317
18  morning 106
19  little 164
20  without 131
21  though 217
22  window 116
23  something 136
24  seemed 242
25  coming 116
26  looked 186
27  friend 166
28  nothing 108
29  things 171
30  moment 106
31  thought 154
32  Arthur 146
33  Professor 172
34  *** Erasing data... ***
35  {}
36  Elapsed time is 0.224086 seconds.
```

**Example #2.** At the prompt, the user provides `txts/84.txt` as a command line argument (line 1). Random entries in the concordance are reported on lines 3-12. Top keys are printed on line 14. The object is cleared and the output message is produced on line 15.

```
 1  python q1.py txts/84.txt
 2  ***** Sample entries: *****
 3  reverie [5573, 6341]
 4  floundering [974]
 5  entertain [6123]
 6  intervened [6399]
 7  lingered [783, 5197]
 8  introduce [1900]
 9  enticements [106, 1311]
10  equals [1262, 3814]
11  everyone [4722]
12  salubrious [1990]
13  ***** Top Keys: *****
14  father 133
15  *+* Erasing data... *+*
16  {}
17  Elapsed time is 0.104677 seconds.
```

**Example #3.** At the prompt, the user provides `txts/foo.txt` as a command line argument (line 1). This argument corresponds to a file that does not exist. The program raises an exception and determines that an error occurred (line 2).

```
1  python q1.py txts/foo.txt
2  ERROR: cannot open txts/foo.txt
3  Elapsed time is 0.000085 seconds.
```

## 2 Important Links

- The Python Standard Library - Built-in Types
- https://www.nltk.org
- Project Gutenberg

## 3 Submitting Your Assignment

Once you have completed your program, submit it (q1.py) in VIU Learn in its corresponding Assignment page.