CS1210 Computer Science I: Fundamentals
Lab 3: Comprehensions

Introduction

In this assignment, you will be writing four short functions, **each involving some sort of comprehension.** Remember:

(1) In the first lab assignment, many of you failed to insert your hawkid (not the same as your student ID number) in the tuple returned by the *hawkid*() function. Because it was the first lab, I corrected all of these by hand. From this point on, however, failure to properly complete the *hawkid*() function will result in a zero grade.

(2) Similarly, please don't forget to complete the *header* of the file (the section of comments at the beginning of the file which should contain your name -- not mine -- and your section identifier). Remember, comments are identified as lines starting with the # character, which are simply ignored by Python.

(3) Please include comments as appropriate.

(4) Each of your functions should return the requested value (printing is not the same as returning).

(5) You are responsible for making sure your solution file loads properly into Python, that is, that there are no mismatched parenthesis, improper indentation, and so on. To this end, it is important that you edit your file in IDLE or another programmer's editor. Files edited in Word, Wordpad, or other non-programmers editors generally don't work and will receive zero credit.

1. Round Up

Write a function *roundUp*(S, r) that takes a sequence, S, of integers and a radix, r, as its arguments and returns a list containing the integers of S rounded up to the next larger multiple of r. So, for example:

```
>>> roundUp(range(3, 12, 2), 2)
[4, 6, 8, 10, 12]
>>> roundUp(range(3, 12, 2), 3)
[6, 6, 9, 12, 12]
>>> roundUp(range(3, 12, 2), 4)
[4, 8, 8, 12, 12]
```

2. Word Ratio

Write a function *wordRatio*(S) that takes a string, S, representing a collection of individual words separated by spaces and returns the ratio of non-space characters to the length of S. So, for example:

```
>>> wordRatio("many of you failed to insert your hawkid")
0.825
>>> wordRatio("supercalifragilisticexpialidocious")
1.0
>>> wordRatio("four score and seven years ago")
0.8333333333333334
```

3. Vowel Count Dictionary

Write a function *vcDict*(S) that takes a string, S, representing a collection of individual words separated by spaces and return a dictionary where each word in S is a key and the number of vowels (defined as a, e,

i, o or u) in that word is the value. So, for example:

```
>>> vcDict("many of you failed to insert your hawkid")
{'to': 1, 'of': 1, 'failed': 3, 'many': 1, 'hawkid': 2, 'your': 2,      \\
 'insert': 2, 'you': 2}
>>> vcDict("supercalifragilisticexpialidocious")
{'supercalifragilisticexpialidocious': 16}
>>> vcDict("four score and seven years ago")
{'score': 2, 'years': 2, 'seven': 2, 'four': 2, 'ago': 2, 'and': 1}
```

Note: the extra \\ at the end of line represent an artificial new line, inserted just to make this handout legible.

4. All Triples

write a function *allTriples*(*k*) that takes an integer, *k*, and returns a tuple containing all triples (x, y, z) such that 0 < x, y, z < k and x > y > z. So, for example:

```
>>> allTriples(3)
((2, 1, 0),)
>>> allTriples(5)
((2, 1, 0), (3, 1, 0), (3, 2, 0), (3, 2, 1), (4, 1, 0), (4, 2, 0),      \\
 (4, 2, 1), (4, 3, 0), (4, 3, 1), (4, 3, 2))
>>> allTriples(0)
()
```

Note: the extra \\ at the end of line represent an artificial new line, inserted just to make this handout legible.

Methods

For paired assignments, you should always login as one of the two partners and work in that partner's directory (in other words, don't try to switch from one machine to the other). In every lab, you will turn in just **one** solution, with both partner's names in it. **Important: you must upload your solution prior to the end of discussion section; no late labs are accepted.**

(1) Download the Python file template provided and save it to your local machine (on the Desktop is fine).

(2) Open the file using IDLE3.

(3) The file contains a special function *hawkid*() along with function *signatures* for each of the functions in the assignment.

(4) Complete the *hawkid*() function according to the instructions given in the template file. **Important: a correct version of this function is required by the autograder to properly credit you for your work.**

(5) Complete each function by replacing the placeholder body of that function with the appropriate expression (you should alternate which partner is "driving" and which partner is "navigating" on each function or at least for every two functions).

(6) You should test each of your functions in the Python REPL.

(7) You should make sure your Python file loads and runs properly; files with syntax or indentation errors will receive no credit.

(8) Upload your Python file to ICON. Note that only the last file updated will be graded. **Important: you need upload only one file to one partner's ICON account; just be sure you include all the partner's names in the** *hawkid*() **function.**

Because we give partial credit, you should consider uploading your Python file to ICON several times (each time making sure the file loads and runs without error) over the course of the lab period to make sure all completed work gets graded.

Finally, don't forget to fill out the lab 3 survey, which is open and available to you until midnight next Monday (September 18). Failure to fill out the survey may result in less-than-full-credit on the assignment.