CS 1301 – Homework 6

FileIO and Dictionary Manipulation

Due: Wednesday October 12th, 11:55pm

Deliverables:

HW6.py

Do **NOT** turn in any other file than HW6.py. There is an output file that your code will generate. If you turn this in, there will be a **10-point deduction**. Turning in a HW file named differently will result in the same deduction.

You should work individually on this assignment. You may collaborate with other students in this class. Collaboration means talking through problems, assisting with debugging, explaining a concept, etc. Students may only collaborate with fellow students currently taking CS 1301, the TA's and the lecturer. You should not exchange code or write code for others. For individual assignments, each student must turn in a unique program. Your submission must not be substantially similar to another student's submission. Collaboration at a reasonable level will not result in substantially similar code.

For Help:

- TA Helpdesk – Schedule posted on class website.
- Email TA's or use Piazza Forums

Notes:

- Do not forget to include the required collaboration statement (outlined on the syllabus).
- Do not wait until last minute to do this assignment in case you run into problems.
- Read the entire specifications document before starting this assignment.

BEFORE YOU BEGIN:

There are 3 functions that you will be required to write. Naming a function differently than as they are outlined will result in NO points for that function.

It is recommended that you start with the first function to fully understand the assignment.

Function: getDict()
Parameters: file (a string representing a filename)
Return: A dictionary

Instructions:

Write a function called getDict() that takes in a filename as a parameter and returns a dictionary. The function should read the file in and create the dictionary from that information. The dictionary should have the following format:

key: Student name represented as a string with a single space separating first and last name.

- Acceptable Keys: "Rachel Golding", "chet shawarma"
- Not Acceptable Keys: "RachelGolding", " chet shawarma "
- Do NOT leave any spaces around the string or multiple spaces between the first and last name. Do not capitalize names if they aren't already.

value: A single list composed of lists.

- Each item in the list is a list representing a class taken by the student
  - Index 0 contains a string representing the class name
    - EG: "CS1301", "CS1332"
  - Index 1 contains a single int (0-100) representing grade
  - Index 2 contains an int representing credit hours
    - Will be a positive number (>0)

See the example file for how the lines will be laid out. Assumptions below on how the line will be laid out will not deviate.

A Sample Dictionary of one person:

{"Rachel Golding": [["CS1301", 100, 3], ["Calculus", 100, 4], ["PhysicsXtreme", 98, 4], [English, 80, 2]]}

Assumptions:
1) You may **NOT** assume that every line has a name. These empty lines, or lines with only spaces, should not be included in the dictionary.
2) Each student and their classes will be separated by a new line character
3) Student names and their classes will be separated by any varying number of spaces and a single colon.

4) Every class will be contained in square brackets "[]" with the 3 items separated by comma.
5) The class items will always be in the same order: Class, Grade, Credit Hours
6) A person might have zero classes.
7) Names will be Last Name, First Name. Change to "First Last" for key
8) Only correct values will be handed in. No need to check for grades > 100 or credit hours < 1. Classes will have grades only from 0 – 100 and credit hours 1 – 5.

Ask for more clarifications on **Piazza**, but your homework will be tested on a file that has an almost identical layout to the one provided with changes made to names and numbers.

You MAY write helper functions and use any additional functions you find online to assist you, but cite other sources in your collaboration statement!
See the last page for the dictionary to the file provided.

Function: calcGPA()

Parameters: student, aDict (a string representing a student and a dictionary from above)

Return: gpa (a float, rounded to two decimals)

Write a function that takes in the dictionary from above to calculate a student's gpa. The function will take in two parameters:
1. **A student**: a string representing a student.
2. **aDict**: a dictionary that may be empty.

Assumptions:
1) The dictionary may be empty. Return 0.0 in this case.
2) The student might have 0 classes. They will have a gpa of 0.0
3) The gpa should range from 0 - 4 (F - A). There will not be values greater than 100 in the dictionary as grades. Convert from 0-100 to 0 - 4.
4) The student might not be in the dictionary. Return 0.0 in this case.

GPA is calculated as follows:
Total Credits: ((credit hours of class1) * (grade1)) + ((credit hours of class2) * (grade2)) + … + [Add up all classes]
gpa = (totalCredits / totalCreditHours)

Note that grade is stored as an int from 0 to 100:

A - 4, B - 3, C - 2, D - 1, F - 0.

Return the gpa as a float. Round to 2 decimal places as needed.
Acceptable return values: 3.0, 3.01, 4.0, 2.13, etc
Not Acceptable: 3, 3.1415926

Function: newFile()
Parameters: classList, aDict, outputFile
Return: None

Parameters:

- classList: The names of all the classes taught by that professor as a list
- aDict: the dictionary made above.
- outputFile: a string representing the name of the output file.

The function will be called like this:

```
>>> aDict = {“Rachel Golding”: [[“CS1301”, 100, 3], [“Calculus”, 100, 4],
[“PhysicsXtreme”, 98, 4], [English, 80, 2]]}
>>> newFile([“CS1301”, “CS1332”], aDict, “out.txt”)
```

Write a function that creates and writes to a new file.
Create a new file with the filename given. The file will have each line starting with the class name followed by a colon and all of the students, separated by commas, that take that class.

Sample out.txt (from the dictionary formed in “text.txt” provided)

```
CS1301:chetie<3 shwarma, charlay bryant, rachel Golding, erica chia, charlie mcknight
CS101:joshua diaddiggoooo
CS:
```

Assumptions:

1. Every line must have the name of the class followed by a colon and the students that have taken that course with a comma after each name
2. You may **NOT** leave a comma after the last name on the line
3. The dictionary will be valid (in order/structure)
4. You must name the file to whatever is passed in as a parameter.
5. A class might not exist in the dictionary. No names should appear because no students took that class.

Final Notes:

- Use Google to help you!
- dictionary.values() and .keys() are not lists. You can turn them into lists.
- Test your code frequently!

Rubric: Note that this rubric deducts points instead of building points.

getDict—Returns Correct Dictionary— **50**

- Extra spaces around key **-5**
- Value is wrong **-15**
- Key is wrong **-10**
- Incorrect Reading from File **-20**

calcGPA—Returns Correct GPA—**25**

- Incorrect Math, Correct Code **-10**
- Cannot Handle Empty Dictionary/Student not in Dictionary **-10**
- Returns string/int instead of float **-5**

newFile —Creates New File with Correct Information—**25**

- Incorrect file name **-5**
- Incorrect line format **-5**
- Students on wrong lines **-10**
- Cannot handle empty class **-5**

Other Deductions:

- Any code outside a function (except imports) **-10**
- Any prints/inputs **-20**
- A syntax error that causes the code to not run on start **-100**

Correct dictionary for the file “text.txt” provided:

```
>>>getDict("text.txt")
{'sam kassem': [['CS9999', 100, 3], ['CS4440', 100, 3], ['CS
3251', 100, 4], ['English', 100, 2]], 'chetie<3 shwarma':
[['CS1301', 100, 3], ['CS2110', 100, 4], ['CS1331', 100, 3],
['CS1332', 99, 3]], 'joshua diaddiggoooo': [['CS101', 99, 3],
['CS 2200', 92, 4], ['PhysicsXtreme', 98, 4], ['Biolgy', 84,
4]], 'charlie mcknight': [['CS1301', 89, 4], ['CS2200', 89, 4],
['CS4400', 89, 4], ['CS3510', 89, 4]], 'erica chia': [['CS1301',
100, 3], ['Calculus', 100, 4], ['PhysicsXtreme', 98, 4],
['English', 80, 2]], 'charlay bryant': [['CS1301', 100, 3],
['LMC3311', 94, 2], ['CS3311', 94, 2], ['CS 1332', 11, 3]],
'rachel Golding': [['CS1301', 100, 3], ['Calculus', 100, 4],
['PhysicsXtreme', 98, 4], ['English', 80, 2]]}
```