**Week 10 Extra In-Class Exercises**

**(more on files)**

**Q1: Grades**

You are given a file called “grades.txt” that contains records of students’ grades of different courses. Each line of the file contains a student’s name, the course he/she took, the term he/she took the course and the grade he/she received for that course. The four columns are separated by the character '#'. Open the file to understand the structure of the file.

**Part (a) [\*\*\*]**

Write a program that creates a new file called 'course\_info.txt' that contains the names of students who received As for each course. If a course doesn’t have any student who received A, that course should not be included in the file. The generated file should look something like the following:

IS101  
Wendy Li  
Joe Wong  
Ng Yin Hui  
Chua Lee Hong  
  
IS111  
Wendy Li  
Ng Yin Hui  
Peter Liu  
  
IS112  
Joe Wong  
Peter Liu  
Eric Wong

Hint: You can consider defining the following helper functions that may be useful for you:

* A function that takes in the name of the course information file as its parameter and returns a list of strings representing all the courses found in the file, each appearing only once in the list.
* A function that takes in the name of a course and the name of the course information file as its parameters and returns a list of students who have received A for that given course.

**Part (b) [\*\*\*]**

Write a program that prints out all pairs of students who have taken the same course together (i.e., during the same term) at least once. If two students have taken more than one course together, they should be displayed only once.

With the file “grades.txt” we provide, your program should display the following information:

Wendy Li, Joe Wong

Wendy Li, Peter Liu

Joe Wong, Peter Liu

Joe Wong, Eric Wong

Ng Yin Hui, Chua Lee Hong

Ng Yin Hui, Eric Wong

Note that the pairs may appear in a different order and also the pair (A, B) is the same as the pair (B, A).

Hint: You can consider defining the following helper functions:

* A function that takes in the name of the course information file and returns a list of strings representing students found in the file, where each student’s name appears only once in the list.
* A function that takes in the name of the course information file and the name of a student as its parameters and returns a list representing the courses that student has taken. Each element of the list is a tuple containing the name of a course and the term in which the student has taken that course.
* A function that takes in two lists, each representing the courses a student has taken. The function returns True if the two lists have at least one element in common and False otherwise.

**Q2: Most Frequent Words [\*\*\*]**

You are given a file that contains a news article. Write a program that reads this file and finds out which **three** words appear most frequently in the article. The program should print out the top-3 most frequent words and their corresponding counts. (You can assume that the article contains at least three different words.)

You should read the file line by line and split each line into individual words by spaces. You should change everything to lowercase before counting the words. You do not need to remove punctuation marks. E.g., if a sentence is “That’s a great idea.” we consider it to have 4 words: “that’s”, “a”, “great”, “idea.” (Note that the last word contains the full stop.)

We’ve provided you with two files: 'article-1.txt' and 'article-2.txt'.

* For 'article-1.txt', the most frequent three words are

the: 46

to: 29

singpass: 18

* For 'article-2.txt', the most frequent three words are

and: 20

the: 16

to: 14

Hint:

* You can use a list [w, c] to store the count c of the word w, where c can be updated while you’re reading the file. You can store the counts of all words as a list of lists. (Note: Next week you’ll learn about dictionaries, which will allow you to solve this same problem with a better solution.)
* After finding the most frequent word from the list that stores each word’s count, to find the second most frequent word, you can repeat the same logic used to find the most frequent word, except that you will skip keep a word if it is the same as the most frequent word.