Assignment 4

Due by 8am Monday, November 23rd, 2015

Read the instructions below carefully. The instructions must be followed. The assignment is worth 5% of your grade. It is out of 30 points (see below for details).

The goal of this assignment is to have the students learn to work with 2D lists, including opening a file and populating a (2D) list with the data from the file; then understanding a given problem, breaking it into smaller subproblems, solving them to finally arrive at the whole solution.

As part of this assignment, I provided a file called `a4_xxxxxx.py`. All of your code has to go inside of that file (except that you will replace xxxxxx in the file name with your student number). The file only has headers of some functions and a few lines of code. Since majority of the code is missing, if you try to run `a4_xxxxxx.py`, it will cause syntax errors. The parts of the code that are provided are not allowed to be changed. Once you are done, submit just that one file containing your solution to Blackboard Learn, i.e. your `a4_xxxxxx.py`.

As usual syntax errors are not allowed. For that reason, if you run out of time and/or your program contains code that causes syntax errors, then comment out that section of code.

Keyword global is not allowed. In particular points will be removed if variables declared as global are used.

1 Description of Assignment 4

1.1 Overview and loose description

Banks lend money to each other. In tough economic times, if a bank goes bankrupt, it will not be able to pay back the loan. A bank's total assets are its current balance plus its loans to other banks. Figure 1 below is a diagram (a graph) that shows five banks. (The figure is just for illustration purposes). The banks' current balances are 25, 125, 175, 75, and 181 million dollars, respectively. The directed edge from node 1 to node 2 indicates that bank 1 lends 40 million dollars to bank 2.

If a bank's total assets are under a certain limit, the bank is unsafe. The money it borrowed cannot be returned to the lender, and the lender cannot count the loan in its total assets. Consequently, the lender may also be unsafe, if its total assets are under the limit. Write a program to find all unsafe banks.

1.2 Detailed description

Your program must follow the format indicated in provided file `a4_xxxxxx.py`. At the minimum it must have the three specified functions and at least one more other function (of your choice).

In addition to `a4_xxxxxx.py` you are also provided with the file called `a4-input.txt`. You should place this file in the same folder (i.e. directory) as `a4_xxxxxx.py`.

In `a4_xxxxxx.py`, in the body of function called `read_initial_info`, your program reads the input from a file `a4-input.txt` (see the docstring of `read_initial_info` for the details of what `read_initial_info` has to do). (To recall how to read a file, generate a list from what is read, etc. revisit the Lec 12 where we did something similar for the quiz game).

While your can test your program on `a4-input.txt` (or `a4-input-second.txt`). Your program has to work on any text file called `a4-input.txt` that follows the following format:

The first line of `a4-input.txt` contains **limit**, that is, the minimum total assets for keeping a bank safe. Each of the remaining lines in the file contains info about one bank. Eg. line 1 contains info about bank with id 0, line 2 about bank with id 1, line 3 about bank with id 2 ... etc. Thus the bank id-s are from **0** to **n-1**, where **n** is the total number of banks (thus the total number of banks is the number of lines in the file minus 1). The first number

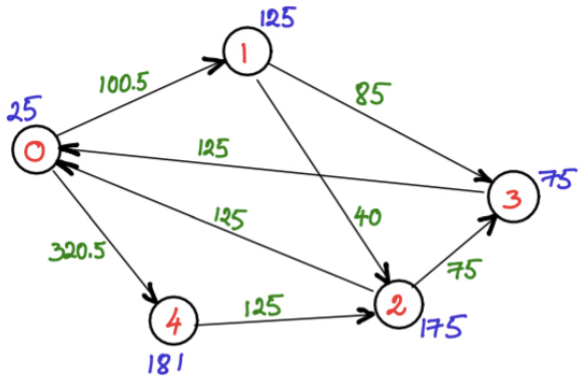

Figure 1:

in the line is the bank's balance, and the rest are pairs of two numbers. Each pair describes a borrower. The first number in the pair is the borrower's id and the second is the amount borrowed.

For example, the input for the five banks in Figure 1 above, is as follows (note that the **limit** is **201**): (The below is also the content of the file `a4-input.txt`)

```
201
25 1 100.5 4 320.5
125 2 40 3 85
175 0 125 3 75
75 0 125
181 2 125
```

For example, for the above input, the total assets of banks are:
Bank 0: Current assets: 446.0
Bank 1: Current assets: 250.0
Bank 2: Current assets: 375.0
Bank 3: Current assets: 200.0
Bank 4: Current assets: 306.0

The bank 3 has assets 75 + 125= 200, which is under 201. So bank 3 is unsafe. After bank 3 is determined unsafe, the total assets of the remaining banks change to:

Bank 0: Current assets: 446.0
Bank 1: Current assets: 165.0
Bank 2: Current assets: 300.0
Bank 4: Current assets: 306.0

Thus the assents of bank 1 change to 125 + 40=160 and consequently fall below the limit. Thus bank 1 is also unsafe. After bank 1 is determined unsafe, the total assets of the remaining banks change to:
Bank 0 Current assets: 345.5
Bank 2 Current assets: 300.0
Bank 4 Current assets: 306.0

None of the remaining banks has assets smaller than the limit, thus we can stop and conclude that the unsafe banks are 1 and 3 and safe banks are 0,2 and 4.

(*Hint*: When you find an unsafe bank, say a bank with id i, you should go through the 2D lists banks and for every bank (i.e. every sublist) that has entry i followed by a number x, you should replace x by zero. Here x represents

the number of millions of dollars that that bank lent to the bank i, as explained earlier. And then recompute the assets of the banks.)

Every time you find one unsafe bank, your program should update the assets of the remaining banks and print their current assets. To understand what exactly is required of your program and what it must display study the two example runs below. As always, the example runs are part of the assignment description and have to he studied to fully understand what is required of your program.

Here is how much various parts of your solution will be worth:

- 10 points - Having your program correctly read the data from any file called a4-input.txt that follows the above explained format, more specifically implementing function `read_initial_info` correctly
- 10 points - Printing the initial assets of all the banks and determining the 1st unsafe bank, if there is one.
- 10 points - determining all unsafe banks correctly, and displaying the updated assets of the remaining banks each time you find one unsafe bank. If there is more than one unsafe bank found in a round. Pick the bank with the smallest ID first. Update the assets of the remaining banks and repeat if necessary (see Example run 2 for more)

2 Example run 1

Here is what your program should print if you click run on `a4_xxxxxx.py` with the provided file `a4-input.txt` in the same folder.

```
>>>
[[25.0, 1, 100.5, 4, 320.5], [125.0, 2, 40.0, 3, 85.0], [175.0, 0, 125.0, 3, 75.0], [75.0, 0, 125.0],
[181.0, 2, 125.0]]


Safe limit is: 201.0 million dollars

Current unsafe banks are:
Current assets of the banks which are not yet in unsafe list:
Bank 0 Current assets: 446.0 millions
Bank 1 Current assets: 250.0 millions
Bank 2 Current assets: 375.0 millions
Bank 3 Current assets: 200.0 millions
Bank 4 Current assets: 306.0 millions


Adding bank 3 to the list of unsafe banks.
Current unsafe banks are: 3
Current assets of the banks which are not yet in unsafe list:
Bank 0 Current assets: 446.0 millions
Bank 1 Current assets: 165.0 millions
Bank 2 Current assets: 300.0 millions
Bank 4 Current assets: 306.0 millions


Adding bank 1 to the list of unsafe banks.
Current unsafe banks are: 3 1
Current assets of the banks which are not yet in unsafe list:
Bank 0 Current assets: 345.5 millions
Bank 2 Current assets: 300.0 millions
Bank 4 Current assets: 306.0 millions

Unsafe banks are:  1 3
Safe banks are:  0 2 4
>>>
```

3 Example run 2

This assignment provides the 2nd text file, called `a4-input-second.txt`. The content of that file is:

```
220
25 1 100.5 4 320.5
125 2 40 3 85
69 0 125 3 75
75 0 125
250 2 125
100 0 80 4 70
150 1 10 2 80 3 40
```

Copy that file into `a4-input.txt` (In other words, imagine that the content of `a4-input.txt` is the same as `a4-input-second.txt`). In that case when you press run on `a4_xxxxxx.py`, your program should print:

```
>>>
[[25.0, 1, 100.5, 4, 320.5], [125.0, 2, 40.0, 3, 85.0], [69.0, 0, 125.0, 3, 75.0], [75.0, 0, 125.0],
[250.0, 2, 125.0], [100.0, 0, 80.0, 4, 70.0], [150.0, 1, 10.0, 2, 80.0, 3, 40.0]]


Safe limit is: 220.0 million dollars

Current unsafe banks are:
Current assets of the banks which are not yet in unsafe list:
Bank 0 Current assets: 446.0 millions
Bank 1 Current assets: 250.0 millions
Bank 2 Current assets: 269.0 millions
Bank 3 Current assets: 200.0 millions
Bank 4 Current assets: 375.0 millions
Bank 5 Current assets: 250.0 millions
Bank 6 Current assets: 280.0 millions


Adding bank 3 to the list of unsafe banks.
Current unsafe banks are: 3
Current assets of the banks which are not yet in unsafe list:
Bank 0 Current assets: 446.0 millions
Bank 1 Current assets: 165.0 millions
Bank 2 Current assets: 194.0 millions
Bank 4 Current assets: 375.0 millions
Bank 5 Current assets: 250.0 millions
Bank 6 Current assets: 240.0 millions


Adding bank 1 to the list of unsafe banks.
Current unsafe banks are: 3 1
Current assets of the banks which are not yet in unsafe list:
Bank 0 Current assets: 345.5 millions
Bank 2 Current assets: 194.0 millions
Bank 4 Current assets: 375.0 millions
Bank 5 Current assets: 250.0 millions
Bank 6 Current assets: 230.0 millions


Adding bank 2 to the list of unsafe banks.
Current unsafe banks are: 3 1 2
Current assets of the banks which are not yet in unsafe list:
Bank 0 Current assets: 345.5 millions
```

```
Bank 4 Current assets: 250.0 millions
Bank 5 Current assets: 250.0 millions
Bank 6 Current assets: 150.0 millions


Adding bank 6 to the list of unsafe banks.
Current unsafe banks are: 3 1 2 6
Current assets of the banks which are not yet in unsafe list:
Bank 0 Current assets: 345.5 millions
Bank 4 Current assets: 250.0 millions
Bank 5 Current assets: 250.0 millions

Unsafe banks are:  1 2 3 6
Safe banks are:  0 4 5
>>>
```