Seneca College

June 08, 2018

Applied Arts & Technology

SCHOOL OF COMPUTER STUDIES

JAC444 **Demo Due dates: June 15 and June 22, 2018**

Final Code Submission Date: June 22, 2018

Workshop 2

Notes:

i. Each task should be presented during the lab, demo worth 50% of the workshop marks and code uploading worth the other 50%.

ii. **At least one task** should be demoed in June 15th lab and the other task should be demoed on June 22nd (Student can choose any task they want to give demo about first).

iii. Make sure you have all security and check measures in place, like exceptional handling and wrong data types etc.

iv. Given output structure is just for student to have a glimpse what the output can look, student are free to make the output better in any way.

Other inputs can be given during demo, so make sure you test your program properly.

Task 1:

Task 1 contains two parts (a) and (b), your program should represent a menu option, like shown below. Once the user will enter its choice the game will run accordingly.

```
Hangman Game

Press 1 to run game with randomly generated word.

Press 2 to run game by reading the word from the text file.

Enter your choice:
```

Part (a)

Let's create a Game called Hangman: Write a hangman game that randomly generates a word and prompts the user to guess one letter at a time, as shown in the sample run. Each letter in the word is displayed as an asterisk. When the user makes a correct guess, the actual letter is then displayed. When the user finishes a word, display the number of misses and ask the user whether to continue to play with another word. Choose data structure of your choice.

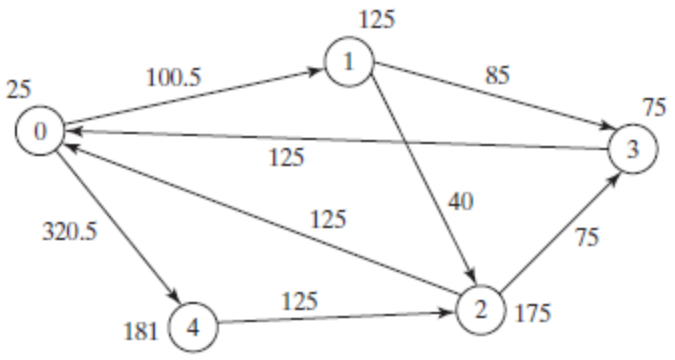
```
(Guess) Enter a letter in word ******* > p
(Guess) Enter a letter in word p****** > r
(Guess) Enter a letter in word pr**r** > p
          p is already in the word
(Guess) Enter a letter in word pr**r** > o
(Guess) Enter a letter in word pro*r** > g
(Guess) Enter a letter in word progr** > n
          n is not in the word
(Guess) Enter a letter in word progr** > m
(Guess) Enter a letter in word progr*m > a
The word is program. You missed 1 time
Do you want to guess another word? Enter y or n>
```

Part (b)

Make the game more interesting, the program reads the words stored in text file name **hangman.txt.** Words are delimited by spaces.

Task 2:

Banks lend money to each other. In tough economic times, if a bank goes bankrupt, it may not be able to pay back the loan. A bank's total assets are its current balance plus its loans to other banks. The diagram below shows five banks. 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 the unsafe banks. Your program reads the input as follows.

1. It first reads two integers **n** and **limit**, where **n** indicates the number of banks and **limit** is the minimum total assets for keeping a bank safe.
2. It then reads **n** lines that describe the information for **n** banks with IDs from **0** to **n-1**.

The first number in the line is the bank's balance, the second number indicates the number of banks that borrowed money from the bank, 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 above picture is as follows (**note that the limit is 201**):

Number of banks: 5
Minimum asset limit: 201
Bank # 0 → Balance: 25 → Number of banks Loaned: 2 → Bank ID: 1 → Amount: 100.5 → Bank ID: 4 → Amount: 320.5
Bank # 1 → Balance: 125 → Number of banks Loaned: 2 → Bank ID: 2 → Amount: 40 → Bank ID: 3 → Amount: 85
Bank # 2 → Balance: 175 → Number of banks Loaned: 2 → Bank ID: 0 → Amount: 125 → Bank ID: 3 → Amount: 75
Bank # 3 → Balance: 75 → Number of banks Loaned: 1 → Bank ID: 0 → Amount: 125
Bank # 4 → Balance: 181 → Number of banks Loaned: 1 → Bank ID: 2 → Amount: 125

The total assets of bank 3 are (75 + 125), which is under 201, so bank 3 is unsafe. After bank 3 becomes unsafe, the total assets of bank 1 fall below (125 + 40). Thus, bank 1 is also unsafe.

The output of the program should be
Unsafe banks are 3 and Bank 1