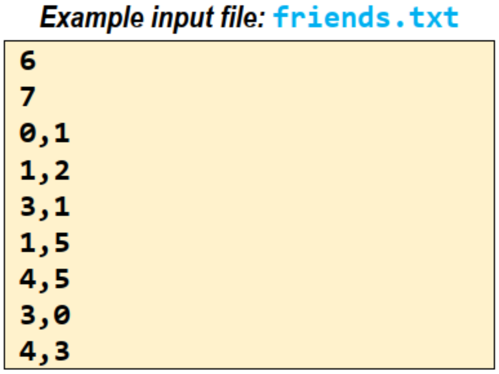
Programming Fundamentals

BDS-(E) Spring 2022

Course Project

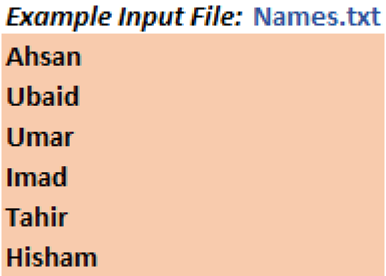
Due :31 May 2022

You are given an input file (**friends.txt**) containing data about friendship status and (**names.txt**) containing personal information about some persons. The first line in the (**friends.txt)** input file contains a positive integer N (you can assume that) indicating the number of persons in a small social network. The second line of this input file contains another positive integer **F** (indicating the number of friendships). After that there are F lines, each line containing a pair of integers indicating that person ' i ' is friends with the person 'j'.



Example input file: friends.txt

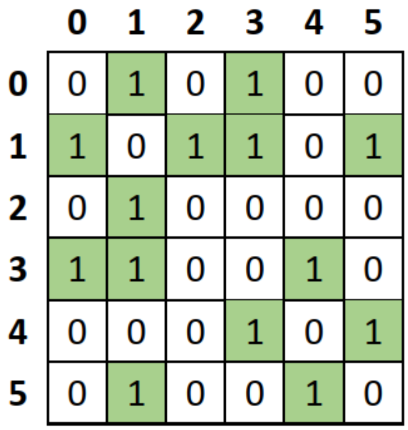
```
6
7
0,1
1,2
3,1
1,5
4,5
3,0
4,3
```



Example Input File: Names.txt

```
Ahsan
Ubaid
Umar
Imad
Tahir
Hisham
```

First of all, you are required to store the friendships in the social network using a $N \times N$ two-dimensional array of integers. In the case of above sample input file, you can visualize the friendships in the form a 6×6 array. Your code must be **generic** i.e. it must work for all values of N in the range $2 \le N \le 100$. For this, you can define a 100×100 two-dimensional array of integers and use only the first N rows and first N columns of this two-dimensional array depending upon the value of N read from the input file. Note that, friendship is a two-way phenomenon. Each pair means that: person i is friend with person j, and also that person j is friend with person i. So, each pair i, j will cause two entries in the two-dimensional friendship array to be set to **1**. For example, the friendships given in the above example text file will be stored like this:



	0	**1**	**2**	**3**	**4**	**5**
0	0	1	0	1	0	0
1	1	0	1	1	0	1
2	0	1	0	0	0	0
3	1	1	0	0	1	0
4	0	0	0	1	0	1
5	0	1	0	0	1	0

Also note that all entries on the main diagonal (which runs from top-left to bottom-right corner) will always be ZERO i.e., a person cannot be friend with himself/herself.

The flow of your program will be as follows:

1. Read the data from the input file and store it in a two-dimensional array (as described above).

2. Display the friendship table on screen, in a neat and readable way. All rows and columns should be properly labeled in the output. (This table should display the **Names** of each person). You are also supposed to save the names of friends against each person in a text file.

3. Repeat the following logic until the user chooses to quit:

   a. Ask the user to enter two different integers (let's say $\boldsymbol{A}$ and $\boldsymbol{B}$).
   Perform input validation to make sure that and are NOT same, and both are valid i.e., $\boldsymbol{0 \le A < N}$ and $\boldsymbol{0 \le B < N}$.
   Note: You MUST use more meaningful variable names.

   b. Your program should determine and display the count and list of all persons which are COMMON FRIENDS of persons $\boldsymbol{A}$ and $\boldsymbol{B}$**.** You are also supposed to write the names of persons that COMMON FRIENDS of $\boldsymbol{A}$ and $\boldsymbol{B}$ in a file "MutualFriends.txt".

   c. Your program should also display Friends suggestions to both $\boldsymbol{A}$ **and** $\boldsymbol{B}$**.** Display the friends and friends of friends of $\boldsymbol{A}$ to $\boldsymbol{B}$ that are not already in the social network of $\boldsymbol{A}$**.** Prompt the user "**Whether he wants to make a friendship**". If "**Yes**" then get the user's choice by showing the available options and update the friend's list against $\boldsymbol{A}$**.** Same is the case with $\boldsymbol{B}$**. (It is mandatory to add new friends in the list). Do not forget to save the status of this newly created friendship in the 'friends.txt' file.**
   e.g., if (4 and 5) are the friends of $\boldsymbol{B}$, (2 and 3) are the friends of $\boldsymbol{A}$, where (7 and 8) are the friends of '**2**' and (1 and 9) are the friends of '**3**' then friend suggestions to **B,** are (1,2,3,7,8, and 9).

4. Show the updated friends list table on console in a neat and readable way.

Instructions:

1. You have to submit the source code with proper comments to elaborate your working.
2. Plagiarism will not be tolerated. It will result in a straight F in the course and forwarded to DC committee, who might award 5 F's in all courses you are taking.
3. The project may be done in a group of two people at max. Include the name and roll number of group members in the comments section of your source code.
4. Late submissions will not be accepted.