Programming Assignment 4
CSci 1132 Data Structures Lab
Spring 2013
Due Date: May 3, 11.59pm ET

The Problem:

In this assignment, you have to model a community of people, and answer queries regarding the community and its member. In the community, every member A is a person that has the following:

- A unique identification number
- A name including first name and last name
- A mother
- A father
- List of friends such that A considers being his/her friends

Note, if A considers B a friend, it does not necessarily follow that B considers A a friend.

Your program should be able to answer the following types of queries:

- For a given person A, find all the children of him/her.
- For a given person A, find all the half-siblings of him/her.
- For a given person A, find all the full-siblings of him/her.
- For a given person A, find the friends of A that consider A as their friend as well.
- For a given person A, find all the persons that consider A as their friend.
- Find who has the most mutual friends.

The precise types of question your program should answer are detailed below.

The Input:

In your program, you should have two input files (community.txt and queries.txt) that shall be input to your program from the command prompt, in that order. This is:

./Assignment4 community.txt queries.txt

The **community.txt** contains the information about the community, where each person has one paragraph, with at least one blank between successive paragraphs. The format of each paragraph of a person is the following:

FIRST NAME: word // should be only one word representing the first name
LAST NAME: word // should be only one word representing the last name
ID: number // should be an unsigned integer representing the Identification Number

FATHER: number // should be any unsigned integer representing the ID of the father
MOTHER: number // should be any unsigned integer representing the ID of the mother
FRIENDS: IDs of this person's friends // a comma-separated list of IDs

Note: The list of FRIENDS contains the IDs of the people that the person in question considers to be his/her friends. Those IDs are not necessarily sorted.

The **queries.txt** file contains a set of queries, one at each line, where each query is one of the following:

FULL-NAME-OF ID
MOTHER-OF ID
FATHER-OF ID
HALF-SIBLINGS-OF ID
FULL-SIBLINGS-OF ID
CHILDREN-OF ID
MUTUAL-FRIENDS-OF ID
INVERSE-FRIENDS-OF ID
WHO-HAS-MOST-MUTUAL-FRIENDS

The output:

Your program should output as many lines as there are query lines, such that the k^{th} output line should be the answer to the k^{th} query line in file **queriesfile.txt**.

In your output line answering a query, should begin by repeating the same query, followed by a colon (:), followed by the query-answer.

For the first three types of query, the query-answer should be the two words representing the first and last name of the answer-person. For the next five types of query, the query-answer should be a list of first name and last pairs, separated by commas, and ordered so that their IDs are in increasing order.

For example, a possible output for **MUTUAL-FRIENDS-OF** 8977 can be:

MUTUAL-FRIENDS-OF 8977: Tom Redman, Merry Smith, Dave Smith

The answer to **INVERSE-FRIENDS-OF** ID query should consist of the people that consider the person with the specified ID to be their friend.

The answer to a **WHO-HAS-MOST-MUTUAL-FRIENDS** query should be the person full name that has the most mutual friends. Please note, in case of a tie, choose the one with the smallest ID.