Project 2: My Tiny Social Network

Discussed: Monday March 9, 2020
Posted: Monday March 9, 2020
Due date: Monday 11:59 pm March 30, 2020

Project Objectives

1. Design and implement a simple user interface.
2. Master concepts on function: prototype, documentation, design, implementation, call.
3. Practice on pointers, pointer array, and dynamic memory allocation.

Project Description

In this project, you will design a simple social network called My Tiny Social Network (MyTSN). Your program will read in the current MyTSN user information from an input file, then provide user a simple interface to manage the MyTSN database. For this project, the input file has only user ID and their friendship information and your interface should support the following operations: (1) search for a user; (2) list a user's friends; (3) add a new friend to a user; (4) add a new user; (5) remove a friendship; (6) remove a user. ***You are required to design and implement functions for each of these operations.***

General Requirements

For this project, your program should run with the following command:

```
a.out user_database.txt update.txt
```

where `a.out` is the compiled executable file, `user_database.txt` is the name of the file that has the information of all the current users of MyTSN, `update.txt` is the database with the updated user information file after the execution of the program. You should do a safety check to ensure that you have the correct number of command line arguments and the files are valid.

Input file:

Each user in MyTSN has a unique 6-digit user ID. Each line in the input file will have one of the following two formats (ignore the <>):

`<UID>#`

`<UID1> <UID2>`

The first one indicates the ID of a user. The `#` sign is not part of the ID, it just indicates the end of an ID. An ID has exactly 6 digits. If it has less than 6 digits, you should pad with 0's. For example, `123` should be printed out both on the screen and output file as `000123`. This also applies to the input file and the user input from the keyboard. Each user must have a line like this to register in MyTSN. The second one indicates a friend relationship between two users. A user may have multiple friends and some users may not have any friends. The size of the input file is unknown.

Output file:

This file stores the information after the updates made by the user during the execution of the program. Use the following format:

```
<UID>#                                  // if the user does not have any friend.
<UID>: <UID1>, <UID2>, ... <UIDk>.
```

Description of the program:

Once the program is running, you should first print out the following main menu:

```
Welcome to MyTSN!

1: search for a user
2: list a user's friends
3: add a new friend to a user
4: add a new user
5: remove a friendship
6: remove a user
0: EXIT

Enter your choice (0-6):
```

Your program will read in the choice (user request), process the request accordingly (see below for the detailed elaboration), and then display this main menu for the next request until user enters `0` to terminate the program.

`1: search for a user`

On this option, your program should prompt the user to enter a user ID and print out one of the following two messages based on whether the user ID is found or not, where `<UID>` is the user ID entered by the user.

```
User <UID> found.
User <UID> not found.
```

`2: list a user's friends`

On this option, your program should first prompt the user to enter a user ID,

```
Enter the user ID:
```

and then print out the user ID of all the friends for this user.

```
User <UID>'s friends: <UID1>, <UID2>, ... <UIDk>.
```

Note that the friend UIDs are separated by a '`,`' followed by a space, except that the last friend's UID will followed by a '`.`'. The k friend UIDs can be listed in any order. It is ok to print out "`friends`" when k=1. If a user does not have any friend, print out

```
User <UID> has no friends.
```

You can assume that the UID is valid (i.e., it consists of only 0-9 and no more than 6 of them), but you need to check whether the UID entered already exists in the MyTSN database. If not, print out the following message and prompt the user to enter another UID.

```
User <UID> does not exist.
Enter the user ID:
```

If the user fails to enter an existing UID in three attempts, your program should bring the user back to the main menu.

3: add a new friend to a user

On this option, your program should prompt the user to enter two user IDs. You can assume that the two UIDs are both valid and different (one cannot be a friend of himself/herself).

```
Enter the user IDs of the two friends:
```

Add in the MyTSN database that these two users have become friends and bring the user to the main menu. If the two users are already friends, print out the following message.

```
Users <UID1> and <UID2> are already friends.
```

4: add a new user

On this option, your program should prompt the user to enter a user ID

```
Enter the user ID:
```

You need to check the UID to ensure that it has only digits 0-9 and no more than 6 digits, and the UID is not already in the MyTSN database. If there is any violation, print out one of the followings.

```
<UID> can only have up to 6 digits.
<UID> already exists.
```

If the user fails to enter a valid UID in three attempts, your program should bring the user back to the main menu.

5: remove a friendship

On this option, your program should prompt the user to enter two UIDs, you can assume that the two UIDs entered are both valid and different.

```
Enter the user IDs of the two friends:
```

Then you need to check whether the two UIDs are friends or not based on the MyTSN database. If they are not, print out the following message and bring the user to the main menu

```
Users <UID1> and <UID2> are not friends.
```

If they are, remove their friendship from the database, print out the following message and bring the user to the main menu.

```
Users <UID1> and <UID2> are no longer friends.
```

6: remove a user

On this option, your program should prompt the user to enter the UID of the user to be removed from the MyTSN database. You can assume that the UID is valid. Then remove the user and his/her friendship information from the MyTSN database, print out the following message and bring the user to the main menu.

```
Users <UID> has been removed from MyTSN.
```

0: EXIT

On this option, your program should write the current MyTSN user information into the output file with the format described earlier, print out the following good-bye message and terminate.

```
Thank you for using MyTSN!
```

Bonus will be given if you can have the output file sorted by UID (in the ascending order) and for the list of friends of each user, also sorted in the ascending order.

Submission Requirements:

1. The project must be implemented using C under the GLUE UNIX system in a file named `p2.c` which should be submitted with the follow command
   `submit 2020 spring enee 150 010x 3 p2.c`
2. Your submission must conform to the expectations outlined above for program execution and input/output files.
3. You must use functions to implement the required operations. Prototype, documentation, and implementation of each function are expected.
4. Code must be reasonably commented to allow a user to follow the program.

Grading Criteria:

Correct compilation
Correct execution
Function prototype and correct implementation
Correct use of pointer and dynamic memory allocation
Coding style and documentation

Files that cannot be compiled will be given a 0.

Late penalty is 20% per day. Submissions more than two days late will not be accepted.