# CENG 301, Spring 2024
## Homework Assignment 2

## Due: 23:59, April 7, 2024

In this homework, you will implement a simple social media platform. The simple social media platform consists of users with unique IDs throughout the system. Users can follow each other. For each user, you will store 2 sorted linked list: one to store pointers to users following that user, the other to store pointers followed by that user.

In your implementation, you MUST use **sorted linked lists** for storing the users.

The simple social media platform will have the following functionalities. The details of these functionalities are given below:

1. Add a new user
2. Remove a user
3. A user follows another user
4. A user unfollows a user
5. Show the list of all users following a user (in sorted order according to their IDs)
6. Show the list of all users (in sorted order according to their IDs)

**Add a user:** The social media platform will allow users to sign up with an ID and a name. Since the user IDs must be unique, the system must check whether or not the specified user ID already exists, and if it exists, it must not allow the operation. The corresponding function will return:

- false if the user with the same ID already exists
- true if the operation is successful

**Remove a user:** The social media platform will allow removing user accounts.The corresponding function will return:

- false if the user with the ID does not exist
- true if the operation is successful

Don't forget to remove the user from its followers lists of followed users etc.

**A user follows another user:** The social media platform will allow users to follow other users. If one or both of the user IDs does not exist, the system must not allow the operation. If the first user is already following the second user, do not perform the operation.

- false if one or both of the user IDs does not exist AND/OR the first user is already following the second user
- true if the operation is successful

**A user unfollows another user:** The social media platform will allow users to unfollow users they follow. If one or both of the user IDs does not exist, the system must not allow the operation. If the first user is not following the second user, do not perform the operation.

- false if one or both of the user IDs does not exist AND/OR the first user is not following the second user
- true if the operation is successful

**Show the list of all users following a user (in sorted order according to their IDs):** The social media platform will allow the display of a list of users following a user in ASCENDING ORDER according to IDs. If the ID does not exist, the system should display a warning message. Output will be printed to console (you can use cout for printing). You should use the output format in the examples. Even a very simple mismatch (including spaces and commas) causes not receiving any points.
Example output 1:
Users following the user with ID 6:
1, Alper
6, Ayse
Example output 2 (if there are no followers of the user):
Users following the user with ID 6:
None
Example output 3 (if user with the ID does not exist):
Cannot show. There is no user with ID 6.

**Show the list of all users (in sorted order according to their IDs):** The social media platform will allow the display of a list of all users' IDs and names. The entries should be in ASCENDING ORDER according to IDs. Output will be printed to console (you can use cout for printing). You should use the output format in the examples. Even a very simple mismatch (including spaces and commas) causes not receiving any points.
Example output 1 (if there is no user in the social media platform):
Users in the social media platform:
None
Example output 2:
Users in the social media platform:
1, Alper
4, Merve
6, Ayse

Below is the headers for `SocialMediaPlatform` , `User`, `LinkedListNode`, `SortedLinkedList` classes that you must write in this assignment. You should implement them in separate .cpp files.

```
//SocialMediaPlatform.h
//DO NOT MODIFY THIS FILE
#ifndef SOCIAL_MEDIA_PLATFORM_H
#define SOCIAL_MEDIA_PLATFORM_H

#include<iostream>
#include<string>

#include "SortedLinkedList.h"

class SocialMediaPlatform {

private:
    SortedLinkedList *users;

public:
    SocialMediaPlatform();
    ~SocialMediaPlatform();

    bool addUser( const int userId, const std::string name );
    bool removeUser( const int userId );

    bool followUser( const int firstUserId, const int secondUserId );
    bool unfollowUser( const int firstUserId, const int secondUserId );
```

```
    void showFollowersOf( const int userId ) const;

    void showAllUsers( ) const;
};
#endif
```

```
//User.h
// You can modify this file if needed, do not change the name of the class

#ifndef USER_H
#define USER_H
#include<iostream>
#include<string>

class SortedLinkedList;

class User {

private:
    int id;
    std::string name;
    SortedLinkedList *followers; // Users who follow this user
    SortedLinkedList *followeds; // Users who are followed by this user

public:
    // define new functions if needed
    // remove existing functions if needed
    User();
    ~User();
    int getId();
    void setId(int id);
    std::string getName();
    void setName(std::string name);
    bool addFollower(User *u);
    bool removeFollower(int userId);
    bool addFollowed(User *u);
    bool removeFollowed(int userId);
    SortedLinkedList *getFollowers();

    friend class LinkedListNode;
    friend class SortedLinkedList;
};
#endif
```

```
//LinkedListNode.h
//You can modify this file if needed, do not change the name of the class
#ifndef LINKED_LIST_NODE_H
#define LINKED_LIST_NODE_H

class User;
class SortedLinkedList;
```

```
class LinkedListNode {
public:
    User *u;
    LinkedListNode *next;

    // define new functions if needed
    // remove existing functions if needed
    LinkedListNode();
    ~LinkedListNode();

    friend class SortedLinkedList;
};

#endif
```

```
//SortedLinkedList.h
//You can modify this file if needed, do not change the name of the class
#ifndef SORTED_LINKED_LIST_H
#define SORTED_LINKED_LIST_H

#include<iostream>
#include<string>

class LinkedListNode;
class User;

class SortedLinkedList {

private:
    LinkedListNode *head;

public:
    // define new functions if needed
    // remove existing functions if needed
    SortedLinkedList();
    ~SortedLinkedList();
    bool add(User *u); // you can change the return type if needed
    bool remove(int userId); // you can change the return type if needed
    bool checkIfExist(int userId); // you may erase this function if needed
    LinkedListNode *getPointerTo(int userId); // you may erase this function if needed
    void print();
    // add more functions if needed
};

#endif
```

Here is an example test program that uses this class and the corresponding output. Your implementation should use the format given in the example output to display the messages expected as the result of the defined functions.

**Example test code:**

```
#include<iostream>
```

```
#include<string>
#include "SocialMediaPlatform.h"

using namespace std;

int main() {

    string truth_table[2] = {"false", "true"};

    SocialMediaPlatform SMP;

    cout<<truth_table[SMP.addUser( 3, "Lonely Artist" )]<<endl;
    cout<<truth_table[SMP.addUser( 2, "Eater Man" )]<<endl;
    cout<<truth_table[SMP.addUser( 4, "Pet Lover" )]<<endl;
    cout<<truth_table[SMP.addUser( 4, "Musician Guy" )]<<endl;
    cout<<truth_table[SMP.removeUser( 4 )]<<endl;
    cout<<truth_table[SMP.addUser( 4, "Cyborg" )]<<endl;
    cout<<truth_table[SMP.addUser( 1, "Gamer" )]<<endl;
    cout<<truth_table[SMP.addUser( 1, "Ali" )]<<endl;
    cout<<truth_table[SMP.removeUser( 7 )]<<endl;
    cout<<truth_table[SMP.addUser( 6, "Sportsman" )]<<endl;
    cout<<endl;

    SMP.showAllUsers();
    cout<<endl;

    cout<<truth_table[SMP.followUser( 1, 6 )]<<endl;
    cout<<truth_table[SMP.followUser( 1, 4 )]<<endl;
    cout<<truth_table[SMP.followUser( 2, 3 )]<<endl;
    cout<<truth_table[SMP.followUser( 2, 4 )]<<endl;
    cout<<truth_table[SMP.followUser( 3, 1 )]<<endl;
    cout<<truth_table[SMP.followUser( 1, 3 )]<<endl;
    cout<<truth_table[SMP.followUser( 1, 3 )]<<endl;
    cout<<truth_table[SMP.followUser( 1, 7 )]<<endl;
    cout<<endl;

    cout<<truth_table[SMP.unfollowUser( 2, 3 )]<<endl;
    cout<<truth_table[SMP.unfollowUser( 2, 6 )]<<endl;
    cout<<truth_table[SMP.unfollowUser( 1, 7 )]<<endl;
    cout<<endl;

    SMP.showFollowersOf(1);
    cout<<endl;
    SMP.showFollowersOf(2);
    cout<<endl;
    SMP.showFollowersOf(3);
    cout<<endl;
    SMP.showFollowersOf(4);
    cout<<endl;
    SMP.showFollowersOf(5);
    cout<<endl;
    SMP.showFollowersOf(6);
    cout<<endl;
    SMP.showFollowersOf(7);
    cout<<endl;
```

```
    return 0;
}
```

**Output of the example test code:**

```
true
true
true
false
true
true
true
false
false
true

Users in the social media platform:
1, Gamer
2, Eater Man
3, Lonely Artist
4, Cyborg
6, Sportsman

true
true
true
true
true
true
false
false

true
false
false

Users following the user with ID 1:
3, Lonely Artist

Users following the user with ID 2:
None

Users following the user with ID 3:
1, Gamer

Users following the user with ID 4:
1, Gamer
2, Eater Man

Cannot show. There is no user with ID 5.

Users following the user with ID 6:
1, Gamer

Cannot show. There is no user with ID 7.
```

**IMPORTANT NOTES:**
**Do not start your homework before reading these notes!!!**

**NOTES ABOUT IMPLEMENTATION:**

1. You ARE NOT ALLOWED to modify the given parts of the header file. You MUST use sorted linked lists in your implementation. You will get no points if you use dynamic or fixed-sized arrays or any other data structures such as vectors/arrays/lists from the standard library.
2. Moreover, you ARE NOT ALLOWED to use any global variables or any global functions.
3. Output message for each operation MUST exactly match the format shown in the output of the example code. Otherwise, you cannot receive points.
4. Using STL (Standard Template Library) is strictly forbidden.

**NOTES ABOUT SUBMISSION:**

1. In this assignment, you must have separate interface and implementation files (i.e., separate `.h` and `.cpp` files) for your class. Your class names MUST BE `SocialMediaPlatform`, `User`, `LinkedListNode`, and `SortedLinkedList`. Your file names MUST BE `SocialMediaPlatform.h`, `SocialMediaPlatform.cpp`, `User.h`, `User.cpp`, `LinkedListNode.h`, `SortedLinkedList.h`, and `SortedLinkedList.cpp` .
2. The code (`main` function) given above is just an example. We will test your implementation using different scenarios, which will contain different function calls. Thus, do not test your implementation only by using this example code. We recommend you to write your own driver files to make extra tests. However, you MUST NOT submit these test codes (we will use our own test code). In other words, do not submit a file that contains a function called `main`.
3. In the "edit" section of VPL, you can run your code (button with rocket sign) with a few example test cases. You can see whether you pass or fail them. We recommend you to use it. It may help you pinpoint the errors in your codes.
4. This assignment is due by 23:59 on Monday, April 7, 2024. You should upload your work to the corresponding VPL activity in ODTUCLass before the deadline. No hardcopy submission is needed.
5. This homework will be graded by your TA **Burak Ferit Aktan** (aktan@ceng.metu.edu.tr). Thus, you may ask your homework related questions directly to him. There will also be a forum on ODTUClass for questions.