Проект

Структури от данни и програмиране

Социална мрежа

Гергана Раденкова
ФН:45480

Глава 1.Увод

Реализирана е програма, която симулира част от функционалностите на една виртуална социална мрежа.

Целта на програмата е да се изпълняват едни от основните функции за всяка социална мрежа. Проектът е реализиран като един клас а в него 4 структури представят данните на социалната мрежа.

Глава 2.Проектиране

За реализацията на класа **SocialNetwork** са използвани помощни структури от STL, с които представянето на социалната мрежа става по-удобно и има по-добра абстракция. Такива структури са vector, unordered_map, pair, queue, string, също така и структурата User и изброения тип friendship_type.

Данните са представени така:

Структурата **User** представя всеки потребител на социалната мрежа с неговото име и години.

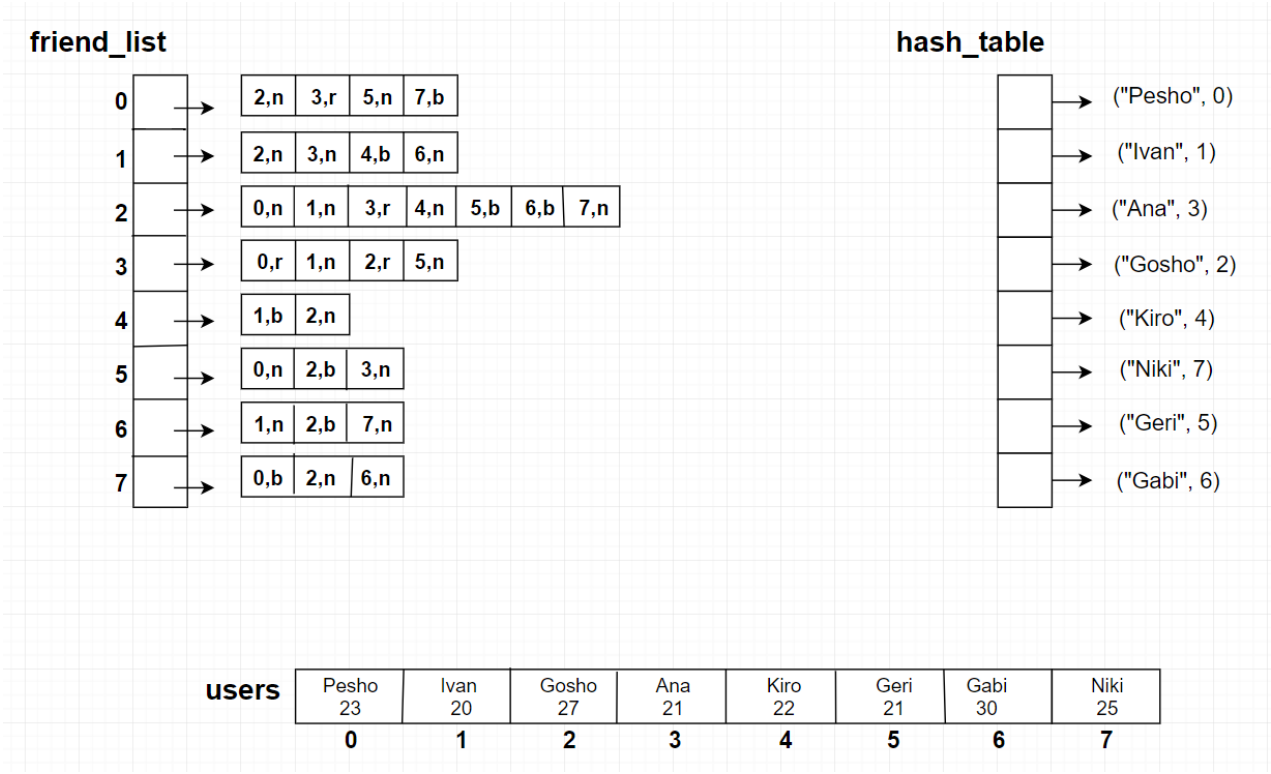
Изброения тип **friendship_type** илюстрира типовете приятелства, които биват: *bestie*, *relative*, *normal* и в допълнение при блокиране на потребител имаме още два вида приятелства *blocked* и *being_blocked*.

std::vector<User*> users – представя потребителите на социалната мрежа в масив. Това ни осигурява бързо търсене по индекс O(1).

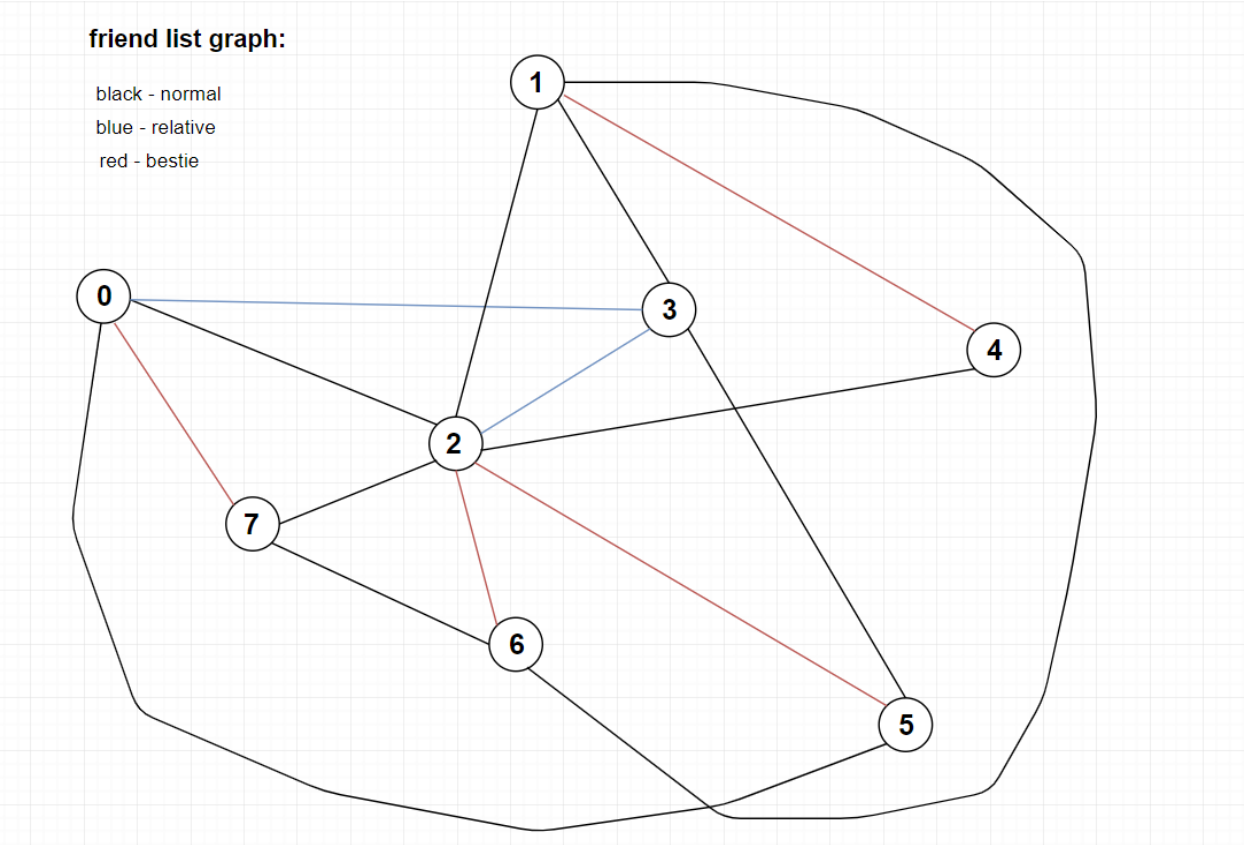
std::unordered_map<std::string, int> hash_table – хеш таблицата предоставя бързо търсене по подаден първи параметър(име на потребител) O(1). А вторият параметър позволява бързо търсене в **users** и **friend_list** O(1).

std::vector<std::vector<std::pair<int, friendship_type>>> friend_list – чрез тази структура от данни представяме приятелствата между потребителите като граф.

std::queue<int> free_positions – при премахване на потребители от социалната мрежа в тази структура от данни записваме техните позиции като свободни. Така при добавяне на нов потребител първо проверяваме за свободни позиции в free_positions.



Графът който представя приятелските връзки в конкретния пример:



Глава 3. Реализация и тестване

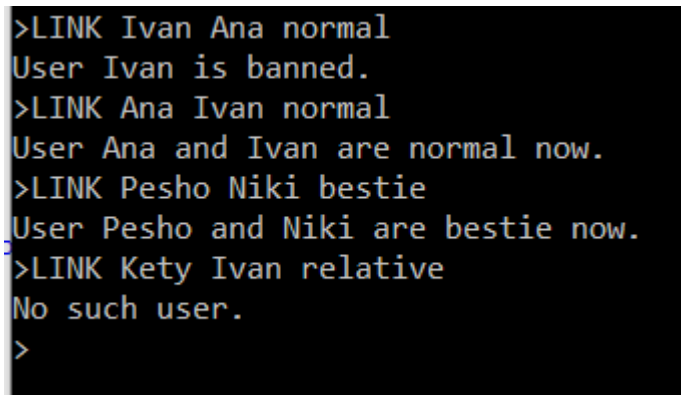
- **CREATE**

create(std::string name, int age) – при успешно добавяне на потребител метода връща истина, ако вече има потребител с това име връща лъжа.

```
>CREATE Pesho 23
User Pesho created.
>CREATE Ivan 20
User Ivan created.
>CREATE Ivan 32
FAIL: Ivan already exists.
>
```

- **LINK**

link(std::string name1, std::string name2, friendship_type type) – ако някой от потребителите не съществува метода връща 1, ако потребителя е баннат връща 2 и при успешно добавяне връща 3.

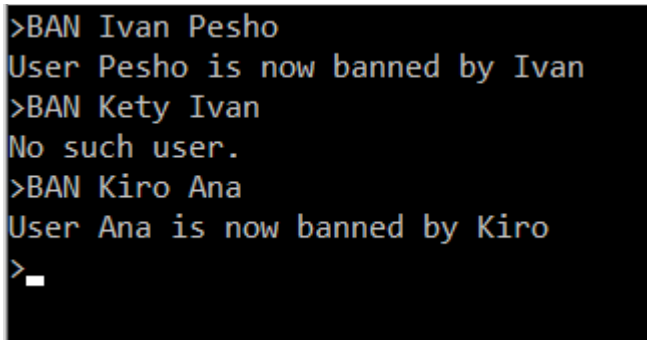


- **FIND**

find(std::string name) – по подадено име на потребител изпечатваме информация за него (име, години, списък с приятели)

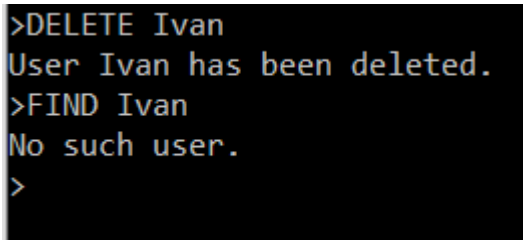
- **BAN**

ban(std::string name1, std::string name2) - метода връща истина, ако банването е било успешно и лъжа когото някой от потребителите не съществува.



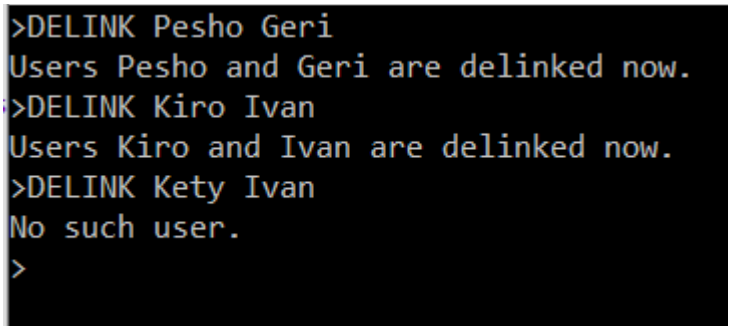
- **DELETE**

remove(std::string name) – ако потребителя не съществува функцията връща лъжа, при успешно изтриване – истина



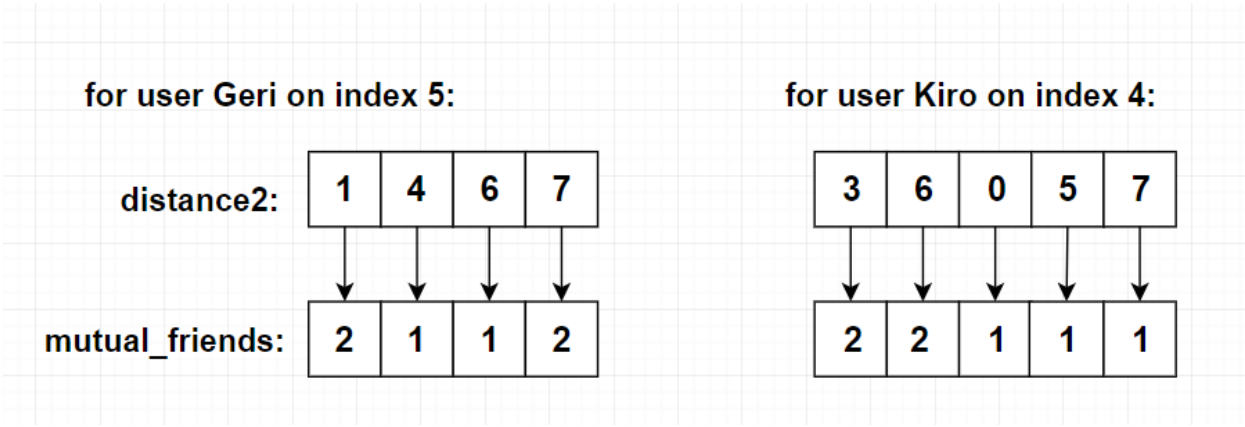
- **DELINK**

delink(std::string name1, std::string name2) – премахва връзката между двамата потребители. При успешно премахната връзка метода връща истина, а ако някой от потребителите не съществува връща лъжа.



- **RECOMMEND**
  - **recommend(std::string name) –** дава списък с предложения за нови приятели на подадения потребител.
  - **get_all_distance2(int start)** – по подаден потребител връща наредена тройка – масив от потребителите на разстояние 2 от подадения потребител, на коя позиция в масива приключват *bestie*-тата, на коя позиция приключват *relative*.
  - **count_mutual_friends(std::vector<int> distance2, int user)** – по подаден масив с потребителите които се намират на разстояние 2, този метод ни дава масив с броя общи приятели на всеки от потребителите от **distance2** и user.

Илюстрация на връзката между двата масива:



- **selection_sort(std::vector<int>& arr1, std::vector<int>& arr2) –** сортира едновременно двата масива от горната картинка, като сортирането се извършва по вектора **mutual_friends**
- **most_popular(int user) –** в случай, че потребителя няма никакви приятели му предлагаме потребителите с най-много приятели.

```
>RECOMMEND Geri
Ivan
Niki
Gabi
Kiro

>RECOMMEND Kiro
Ana
Gabi
Pesho
Geri
Niki

>CREATE Kety 23
User Kety created.
>RECOMMEND Kety
Gosho
Ivan
Pesho
Ana
Geri
Gabi
Niki
Kiro
```

- **SAVETOFILE –** записва данните за социалната мрежа във файл, като края на всяка отделна структура се оказва с думата end в файла.
- **READFROMFILE –** изчита данните за социалната мрежа от файл.
- **EXIT –** изход от програмата.

Глава 4. Заключение

Програмата поддържа основни операции за една социална мрежа – добавяне и премахване на потребител, създаване на приятелство между двама потребители, премахване на приятелство, блокиране на потребител, предложения за нови приятелства. Класът SocialNetwork може да бъде разширен с още функционалности, може да се добавят още видове взаимоотношения.

Референция към github:

https://github.com/GeryRadenkova/Data-Structures/tree/master/Social_Network_Project
https://github.com/GeryRadenkova/Data-Structures/tree/master/Social_Network_Project