Социална мрежа- Документация

Увод

Целта на този проект е да пресъздаде на много базово ниво прилложенията на социална мрежа в термините на езика C++.

Основни цели и задачи:

1.Създаване на потрбител

2.Създаване на приятелство

3.Премахване на приятелсво

4.Баннване

5.Премахване на потребител

6.Търсене на потребител

7.Предложение за приятелства на потребител

Обща архитектура

Проекта е изграден на 2 основни класа: User и Social Media

Целта на класа User е да описва характеристикете които един потребител има:

1.Име

2.Години

3.ID – уникално число с което идентигицираме всеки един потребител

Целта на класа Social Media е да симулира как една социална мрежа работи.Това става чрез

използването на неориентиран тегловен граф използваики vector<vector<pair<User,int>>> graph.

· Vector<User> allUsers е създаден с цел да съдържа всичи потребители в едно

Множество като в него пазим само и единствено потребители без информация дали

имат приятели.

Интерфеисът на класа се състои от следните методи:

· bool AddUser(User& temp) ->добавя потребител в allUsers, но ако се опитаме да добавим потребител с някое име което го има в AllUsers просото няма да го добави.

· int ReturnCommand(const string& command);-> по подаден тип приятеслтво ,

които са “normal”,”bestie”,”family”,връща число с което ще покажем каква ще е тежестта на определено ребро между 2 върха(потребителя).Ако приятелството е „normal“ връща 1 , ако е „bestie“ връща 2 , ако е “family“ връща 3. 4 служи

за показване че 2 потребителя са баннати.

· bool MakeFriendShip(const string& user1, const string& user2,string typeOfFriendShip="normal");->по подадени 2 имена и тип на приятество ги добавя в графа.Първо проверяваме дали тези 2 потребителя съществуват в AllUsers и ако

ги намерим чак тогава ги добавяме в графа , в противен случеи съобщаваме че се опитваме да свържем потребил който не съществува.Начинът по който правим приятелство е следният:Взимаме id на потрбител 1 и потребител 2 и казваме така(псевдо код):graph[idUser1].push_back(//info about User2)

graph[idUser2].push_back(//info about User 1.

· bool RemoveUser(const string& name);->по подадено име премахва потребител от allUsers и от графа.

· void InfoForSearchedUser(const string& name);->по подадено име връща информация

за потребителя във формат :Име , годинин и списък от неговите приятели.

Начинът по който намира списъка от приятелите е следния:Взимаме id то на потребителя и изкармане на конзолата информацията за graph[idUser] тъй като там се намират неговите приятели

- void Ban(const string& name1, const string& name2);->по подадени 2 имена банваме 2 потребителя като сменяме тежестта на реброто им на 4.Аналогично

взимаме id тата на двата потребителя и намираме потребителя и променяме тежестта на 4(пример в 1та посока :на graph[idUser1] намираме потребитля(User2) който искаме да баннем и променяме теяестта, анлогично за graph[idUser2])

- void RemoveFriendShip(const string& name1, const string& name2); -> по подадени 2 имена премахваме от графа приятелството на 2та потребителя.Начинът по който ще стане е подобен с id тата.Намираме на коя позиция в graph[idUser1] се намира User2(нека е i) и правим graph[idUser1].erase(graph[idUser1].erase(graph[idUser1].begin()+i))(тук трием приятелството на User1 s User2),аналгично в другата посока за User2 като искаме да изтрием приятеслтвото му с User1

- void WriteToFileUsers(const string& filename); ->по подадено име на фаил записваме графа във фаил.Записването става по следния начин.Първо във фаила записваме инфорамцията която имаме във allUsers във формат : id name age

ш

всяко на нов ред.После записваме приятелствата от графа.Начинът по който става е така: записваме приятеслтсвата във формат id1 id2 всяко на нов ред.

Този начин означава потребител с id1 е приятел с потребител с id2 .(ще има ред id2 id1).

Причината по която имаме vector<int> indexOfReomvedUser; е че когато сме премаханли постребитрел от мрежата ние му запазмаме id то в този indexOfReomvedUser и когато пишем във фаила по id та ,когато видим например че във indexOfReomvedUser има индекси 3 и 5 то правим следната проверка (начинът по който го пищем е следният въртим цикъл до размера на графа,нека i е променливата от цикъла) дали на графа на позиция i е празен т.е. няма приятелства които да пищем и дали i съвпада със някои индеск от indexOfReomvedUser.Ако някое от двете условия е изпълнено то не записваме вув фаила

- void RestoreGraph(const string& filename);->възстановява множеството allUsers и графа.Това става с външна функция isEveryThingANumber(string line)Тази функция връща като резултат true ако прочетената линията код е описва приятелство(0 1 3 (потребител с id 0 е приятел с потребитрл с id 1 от тип family)) и връща false ако линията код описва потребител(например 0 Petko 14)

· void recommend(const string& name);-> предлага приятели на потребител.

Предлагането става на следният приинцип: Взимаме всички приятели на този потребител и после взимаме всички приятели на тези приятели и пресмятаме коефициентите и ако намерим 30 потребителя прекратяваме,в противен слущеи минаваме ниво надолу(т.е. приятелите на приятелите на приятелите) и така докато не стигнем 30 потрбителя да предложим или не обходим целия графа (използваме bfs)