МІНІСТЕРСТВО ОСВІТИ I НАУКИ УКРАЇНИ

НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ УКРАЇНИ

«КИЇВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ   
ІМЕНІ ІГОРЯ СІКОРСЬКОГО»

ФАКУЛЬТЕТ ІНФОРМАТИКИ ТА ОБЧИСЛЮВАЛЬНОЇ ТЕХНІКИ

СПЕЦІАЛЬНІСТЬ 126 «ІНФОРМАЦІЙНІ СИСТЕМИ ТА ТЕХНОЛОГІЇ»

**КУРСОВА РОБОТА**

з дисципліни «***Структури даних та алгоритми***»

Варіант №6

|  |  |  |
| --- | --- | --- |
| **Керівник :**  доц.Крамар.М., |  | **Виконав:**  студентка гр. ІС-02, ФІОТ  Павлущенко О.В.  залікова книжка № ІС-0222 |
| Допущено до захисту |  |
| І\_\_\_І \_\_\_\_\_\_\_\_\_ 2021 \_\_\_\_\_\_\_\_\_  підпис |  |
| Захищено з оцінкою  \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_  оцінка підпис  І\_\_\_І \_\_\_\_\_\_\_\_\_ 2021 |  |

Київ-2021

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Факультет | ***Інформатики та обислювальної техніки*** | | | | |
|  | (назва факультету, інституту) | | | | |
| Кафедра | ***Автоматизованих систем обробки інформації і управління*** | | | | |
|  | (назва кафедри) | | | | |
| Дисципліна | ***«Структури даних та алгоритми»*** | | | | |
|  | (назва) | | | | |
| Курс | ***1*** | Група | ***ІС-02*** | Семестр | ***2*** |

**ЗАВДАННЯ**

**на курсовий проект (роботу) студента**

|  |
| --- |
| ***Павлущенко Ольги Вікторівни*** |
| (прізвище, ім’я, по батькові) |

.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 1. Тема роботи: | | ***Розробка програмного забезпечення з використанням*** | | | |
| ***об’єктно-орієнтованого підходу.*** | | | | | |
|  | | | | | |
|  | |  | | | |
| 2. Строк здачі студентом закінченого проекту (роботи) | | | | | ***20.05.2021*** |
|  | |  | | | |
| 3. Вихідні дані до проекту (роботи): | | | | ***Варіант №6. Соціальна мережа.*** | |
| ***Комунікації між членами цієї мережі*** | | | | | |
|  | | | | | |
|  | | | | | |
|  | |  | | | |
| 4. Зміст розрахунково-пояснювальної записки (перелік питань, які підлягають | | | | | |
| розробці): | ***Розробка алгоритму роботи програми, архітектура програми.*** | | | | |
|  | | | | | |
|  | | | | | |
|  | |  | | | |
| 5. Перелік графічного матеріалу (з точним зазначенням обов'язкових креслень): | | | | | |
| ***діаграми класів в нотації UML*** | | | | | |
|  | | | | | |
|  | |  | | | |
| 6. Дата видачі завдання: | | | ***1.04.2021*** | | |

|  |  |  |  |
| --- | --- | --- | --- |
| **№п/п** | **Назва етапів курсового проекту (роботи) та питань, які мають бути розроблені відповідно до завдання** | **Термін виконання етапу** | **Позначки керівника про виконання завдань** |
| 1. | Отримання завдання на курсову роботу | 01.04.2021 |  |
| 2. | Підготовка ТЗ | 08.04.2021 |  |
| 3. | Аналіз предметної області | 15.04.2021 |  |
| 4. | Проектування архітектури програмної системи | 21.04.2021 |  |
| 5. | Розробка сценарію роботи програми | 21.04.2021 |  |
| 6. | Узгодження з керівником інтерфейсу користувача | 25.04.2021 |  |
| 7. | Розробка програмного забезпечення | 01.05.2021 |  |
| 8. | Узгодження з керівником плану тестування | 07.05.2021 |  |
|  | Тестування програми | 08.05.2021 |  |
| 9. | Підготовка пояснювальної записки | 11.05.2020 |  |
| 10. | Здача курсової роботи | 20.05.2020 |  |

**КАЛЕНДАРНИЙ ПЛАН**

**Студент \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

(підпис)

**Керівник \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**  Павлущенко Ольга Вікторівна

(підпис) (прізвище ім'я, по батькові)

**«\_\_\_\_» \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ 2021 р.**

**Зміст**

[**Вступ** 5](#_Toc72421807)

[**Аналіз предметної області** 6](#_Toc72421808)

[**1.1 Опис предметної області** 6](#_Toc72421809)

[**1.2 Функціональність програми** 7](#_Toc72421810)

[**Розділ 2. Опис програмного забезпечення** 8](#_Toc72421811)

[**2.1 Опис логічної структури** 8](#_Toc72421812)

[**2.2 Проектування програмного забезпечення** 14](#_Toc72421813)

[**Розділ 3. Інструкція користувача** 15](#_Toc72421814)

[**3.1 Робота з програмою** 15](#_Toc72421815)

[**Розділ 4. Тестування** 18](#_Toc72421816)

[**4.1 Опис тестування програми** 18](#_Toc72421817)

[**4.2 Порядок тестування програми** 18](#_Toc72421818)

[**Висновки** 21](#_Toc72421819)

[**Список використаних джерел** 22](#_Toc72421820)

# **Вступ**

Основною метою курсової роботи з дисципліни «Структури даних та алгоритми» є закріплення на практиці вміння використовувати основні концепції об'єктно-орієнтованого підходу (ООП) — класи, інкапсуляцію, успадкування, поліморфізм, шаблони методів і класів, бібліотек.

Темою даної роботи була реалізація соціальної мережі, встановлення комунікації між її членами. Надання можливості користувачу обирати друзів, надсилати запрошення їм на дружбу, приймати запрошення від інших користувачів. Для реалізації даних завдань було реалізовано основні класи: юзер, в якому визначені основні дії користувача та Відносини, що описує вид відносин між користувачами.

Основним функціоналом має бути:

1. Вхід у свій аккаунт
2. Перегляд друзів
3. Перегляд отриманих запрошень
4. Обробка запрошення
5. Надіслати запрошення іншому користувачеві
6. Вихід із мережі з можливістю входу іншого користувача

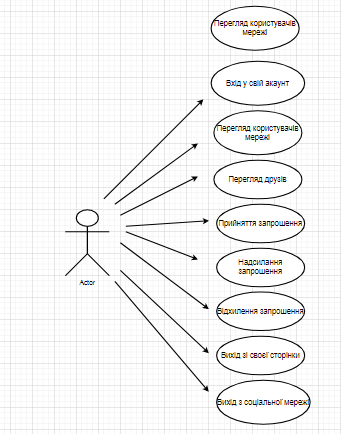
Програма повинна забезпечити можливості комунікації між членами мережі та зробити процес користування мережею зручною для учасників.

# **Аналіз предметної області**

## **1.1 Опис предметної області**

Дана предметна галузь описує взаємодію користувачів в соціальній мережі.

Принципи ООП дозволяють організувати структуру відносин, категорії користувачів. Програма повинна забезпечитися зручне користування мережею, встановлення виду відносин між користувачами, можливість переглянути користувачів, можливість змінювати вид відносин, за рахунок запрошення стати другом чи можливості запросити іншого користувача до своїх друзів. Програма має показувати друзів користувача. Доступне і просте меню дозволить користувачеві розуміти можливості виконання своїх дій.

Таким чином, об’єктами предметної області є користувачі та відносини між ними.

## **1.2 Функціональність програми**

Програма повинна надати можливість користувачеві ввійти у свій аккаунт соціальної мережі, шляхом вводу свого логіну. Після цього, користувач має побачити інших зареєстрованих користувачів, для пошуку нових можливих друзів. Дана програма має забезпечити перегляд друзів активного користувача, перегляд запрошень та обробку їх: прийняття або відхилення; надсилання запрошення стати другом іншому користувачеві. Також має бути можливість вийти зі свого аккаунта та ввійти іншому користувачеві, при цьому статус відносин має зберегтись. Це все має бути забезпечене користувацьким меню, команди якого обирає активний юзер.

Дані зберігаються у вигляді списків зареєстрованих користувачів з даними про них та стану відносин між ними. Користувач, який не ввійшов в свій аккаунт має обмежений список дій. Тому має забезпечитися вхід користувача і розширення функціоналу для нього. Для цього повинен бути створений клас залогіненого юзера, що повинен містити основні методи взаємодії зі своїм аккаунтом та іншими користувачами. Для регулювання відносин потрібен ще один клас, який забезпечує перевірку стану відносин між користувачами, не дає можливості повторно надіслати запрошення одному і тому ж користувачеві, надіслати запрошення користувачам, які вже є знаходяться у відносинах дружності з активним юзером. Тому потрібні методи, які визначають друзів користувача, стан запрошень. Також потрібен абстрактний клас користувача, що має ID та ім’я користувача, також метод перегляду користувачів. Класи-наслідники залогінений користувач та незалогінений отримують інформацію про юзера та власні методи, що визначають ступінь можливості діяльності користувача в межах мережі.

# **Розділ 2. Опис програмного забезпечення**

## **2.1 Опис логічної структури**

Дана програма складається з 8 класів. Клас Program взаємодіє з іншими класами. Являє собою акаунт користувача з діями, який другий може виконати.

Клас «LoggedUser»

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| № | Сигнатура | Вхідні параметри | Повернене значення | Призначення |
| 1. | ShowUsers (List<LoggedUser> RegisteredUsers, int LoggedUserId) | Список зареєстрованих користувачів, id активного користувача | Список користувачів | Метод, що показує зареєстрованих користувачів |
| 2. | AddFriend(List<Friendship> RelationList, List<LoggedUser> RegisteredUsers, int IdUser, int IdReciver) | Список, що містить дані про відносини користувачів, список зареєстрованих користувачів, id активного користувача та отримувача запрошення | Дані користувача, якому надіслано запрошення | Надсилання запрошення на дружбу іншому користувачу |
| 3. | GetUserInvitations(List<Friendship> RelationList, int LoggedUserId) | Список відносин користувачів, id активного користувача | Список запрошень, надісланих користувачу | Отримання запрошень користувача |
| 4. | Accept(List<LoggedUser> RegisteredUsers, LoggedUser loggedUser, int SenderId, List<Friendship> RelationList) | Список відносин, зареєстрованих користувачів, id надсилача повідомлення | Ім’я нового друга | Приймання запрошення на дружбу від іншого користувача |
| 5. | LoggedUser Decline(List<LoggedUser> RegisteredUsers, LoggedUser loggedUser, int SenderId, List<Friendship> RelationList) | Список зареєстрованих користувачів, відносин, id надсилача повідомлення | Ім’я відхиленого користувача | Відхилення запрошення на дружбу |
| 6. | FindName(List<LoggedUser> RegisteredUsers, LoggedUser NewFriend) | Список зареєстрованих користувачів, відносин, користувача | Ім’я користувача | Пошук в списку зареєстрованих користувачів інформації про потрібного користувача |
| 7. | LogOut(LoggedUser loggedUser) | Обєкт користувач | Інформація про користувача | Вихід користувача із акаунта |
| 8. | ShowUserFriend(LoggedUser LoggedUser, List<Friendship> RelationList, List<LoggedUser> RegisteredUsers) | Список відносин, зареєстрованих користувачів, користувач | Список друзів користувача | Отримання друзів активного користувача |
| 9. | public int GetId(List<LoggedUser> RegisteredUsers, string UserName) | Список зареєстрованих користувачів, ім’я потрібного користувача | Id користувача | Отримання Id користувача |

Містить методи, що описують можливості юзера отримання інформації

**Клас «UnloggedUser»**

Містить методи користувача, який не ввійшов в свій акаунт.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| № | Сигнатура | Вхідні параметри | Повернене значення | Призначення |
| 1. | ShowUsers (List<LoggedUser> RegisteredUsers, int LoggedUserId) | Список зареєстрованих користувачів, id активного користувача | Список користувачів | Метод, що показує зареєстрованих користувачів |
| 2. | LogIn(List<LoggedUser> RegisteredUsers, int IdNewUser) | Список зареєстрованих користувачів, id користувача, що хоче ввійти | Дані користувача, якому надіслано запрошення | Надсилання запрошення на дружбу іншому користувачу |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| № | Сигнатура | Вхідні параметри | Повернене значення | Призначення |
| 1. | Relations(List<Friendship> friendships) | Список відносин |  | Констуруктор з параметром, заповнює поле переданим значенням |
| 2. | HaveRelation(int UserId) | id активного користувача | Список користувачів | Метод отримання користувачів, з якими в активного юзера є відносини |
| 3. | List<Friendship> GetFriends(int UserId, List<Friendship> RelationList) | id активного користувача, список відносин | Список друзів | Список користувачів, які є друзями активного користувача |
| 4. | GetRelationType(int SenderId, int ReceiverId, string Name) | Список відносин, зареєстрованих користувачів, id надсилача, id отримувача | - | Приймання запрошення на дружбу від іншого користувача |

**Клас «Relations»**

Містить методи обробки відносин між користувачами

**Клас «Friendship»**

Містить методи обробки інформації двох користувачів, які перебувають у відносинах

|  |  |  |  |
| --- | --- | --- | --- |
| № | Сигнатура | Повернене значення | Призначення |
| 1. | GetStatusText() | Тип відносин | Відношення між двома користувачами |
| 2. | GetSenderId() | id надсилача | Метод отримання id надсилача |
| 3. | GetRecipientId() | id отримувача | Метод отримання id отримувача |
| 4. | IsStatusFriends() | Логічне значення true/false | Метод встановлення чи є користувачі друзями |

**Клас «User»**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| № | Сигнатура | Вхідні параметри | Повернене значення | Призначення |
| 1. | int Id { get; set; }  string Name { get; set; } | - | - |  |
| 2. | ShowUsers(List<LoggedUser> RegisteredUsers, int LoggedUserId) | Список зареєстрованих користувачів, id користувача, що хоче ввійти | Список зареєстрованих користувачів | Список зареєстрованих користувачів |

Абстрактний клас, який визначає задання інформації про користувача та метод показу всіх зареєстрованих користувачів

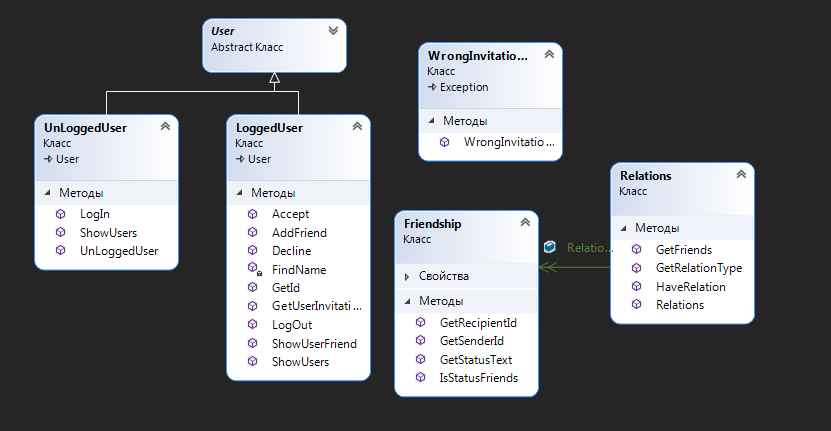
**Клас «FrontLib»**

Містить методи, що встановлють, яким чином показувати інформацію користувачу.

Методи:

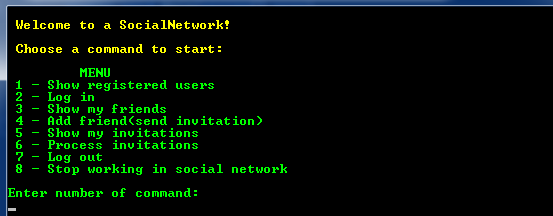
* ShowUsers() – показ користувачів;
* LogIn() – вхід користувача;
* ShowUserFriends() – показ друзів активного користувача;
* SendInvitation() - надсилання іншому користувачеві запиту на дружбу;
* ShowUserInvitations() - показ запрошень активного користувача;
* ProcessInvitation() – обробка запрошень активного користувача;
* AcceptInvitation() – прийняття запрошення;
* DeclineInvitation() – відхилення запрошення;
* Logout() - вихід зі свого акаунта;

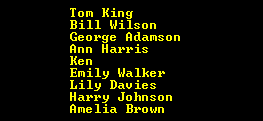
## **2.2 Проектування програмного забезпечення**

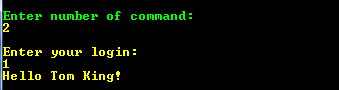


# **Розділ 3. Інструкція користувача**

## **3.1 Робота з програмою**

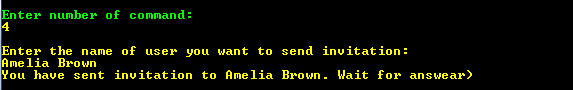
Програма починається з висвітлення меню доступних дій для користувача та можливістю вибору команди.

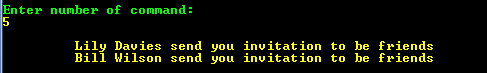
Переглянути користувачів можна обравши пункт 1

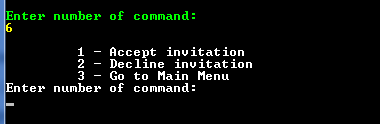
Користувач може ввійти у свій акаунт, обравши пункт 2 та ввівши свій логін.

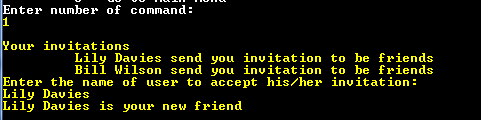
Даний користувач стає активним і для нього доступні інші можливості програми

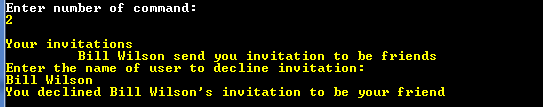
Користувач може переглянути список своїх друзів, обравши команду 3

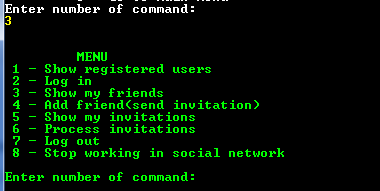
Користувач може надіслати іншому зареєстрованому користувачеві запрошення стати другом, обравши команду 4 та ввівши ім’я отримувача

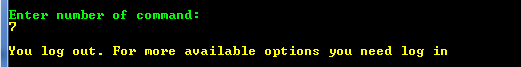
Користувач може побачити, хто надіслав йому запрошення, обравши команду 5

Користувач може обробити запрошення, які йому були надіслані. Обравши команду 6, йому відкривається додаткове меню обробки запрошень.

Для прийняття запрошення потрібно обрати команду 1, після чого висвітлюється список запрошень. Далі, для прийняття запрошення вводится ім’я користувача, що надіслала його. Після чого висвітлюється повідомлення успішного додавання до своїх друзів

Для відхилення запрошення потрібно обрати команду 2, після чого висвітлюється список запрошень. Далі, для прийняття запрошення вводится ім’я користувача, що надіслала його. Після чого висвітлюється повідомлення про відхилення

При виборі пункту 3 виконується перехід до головного меню

Користувач може вийти з акаунту, обравши пункт 7

# **Розділ 4. Тестування**

Зазвичай найбільше помилок у подібних програмах виникає при вводі даних від користувача, тому для даного тестування бажано використати функціональне тестування, тобто аналізувати функціональні характеристики додатка та перевіряти на невідповідності між реальною поведінкою реалізованих функцій та очікуваною поведінкою відповідно до сценарію.

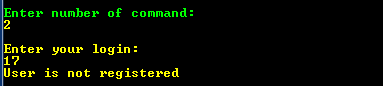
## **4.1 Опис тестування програми**

Тестування буде проводитися з використанням будь-яких вхідних даних від користувача, щоб передбачити усі помилки, які можуть бути можливі при використанні даної програми.

## **4.2 Порядок тестування програми**

1. Тестування входу в аккаунт
2. Тестування введення невірного введення даних користувачів
3. Тестування запитів виконання команд користувачем, що не увійшов у свій аккаунт недоступних для нього дій

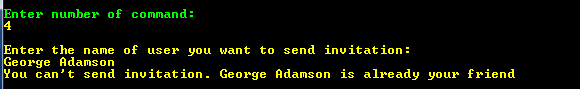
1.



1. Намагання увійти в аккаунт незареєстрованого користувача

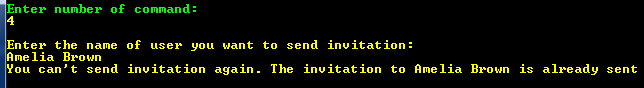


1. введення некоректних даних

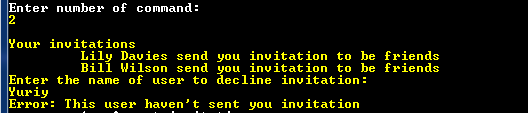
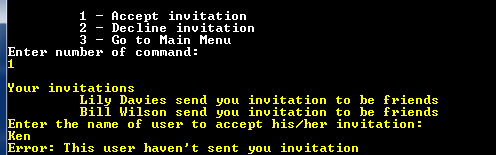


1. надсилання запрошення користувачу, який вже є другом

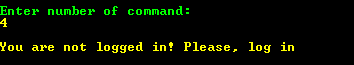
2.



1. надсилання запрошення користувачу, якому вже було надіслано



1. намагання прийняти чи відхилити запрошення від користувача, який не надсилав запрошення чи якого взагалі не існує

3.

Намагання виконати дії недоступні для користувача, що не ввійшов у свій аккаунт

# **Висновки**

Головною задачею даної курсової роботи було реалізувати програму, що забезпечить коректну взаємодію користувача зі складом шляхом складання замовлень та їх відправки. Було розглянути чотири основних пункти:

* Аналіз предметної області
* Опис програмного забезпечення
* Інструкція користувача
* Тестування роботи програми

Основним завданням було створити соціальну мережу, користувачі якої можуть взаємодіяти між собою, керувати відносинами.

Під час тестування критичних ситуацій виявлено не було, адже взаємодія з кодом мінімальна, а всі помилки, які могли б виникнути при вводі даних були відновлені. Програма може бути вдосконалена новими можливостями для користувача.

Отже, під час написання даної роботи були розвинуті та вдосконалені навички використання різних підходів ООП, що і було метою курсової.

# **Список використаних джерел**

1. Програмний код [ЕЛЕКТРОННИЙ РЕСУРС] - <https://github.com/olha2411/SocialNetwork>
2. Полное руководство по языку программирования С# 9.0 и платформе .NET 5 [ЕЛЕКТРОННИЙ РЕСУРC] – режим доступу до ресурсу: <https://metanit.com/sharp/tutorial/>
3. Объектно-ориентированное наследование [ЕЛЕКТРОННИЙ РЕСУРC] – режим доступу до ресурсу: <https://docs.microsoft.com/ru-ru/dotnet/csharp/tutorials/inheritance>
4. C# 4.0: полное руководство. Шилдт Герберт
5. CLR via C#. Программирование на платформе Microsoft .NET Framework 2.0 на языке C#. Джеффри Рихтер.