МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ

ВІДОКРЕМЛЕНИЙ СТРУКТУРНИЙ ПІДРОЗДІЛ

«ПОЛТАВСЬКИЙ ПОЛІТЕХНІЧНИЙ ФАХОВИЙ КОЛЕДЖ

НАЦІОНАЛЬНОГО ТЕХНІЧНОГО УНІВЕРСИТЕТУ

«ХАРКІВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ»

Циклова комісія дисциплін програмної інженерії

**КУРСОВА РОБОТА**

з дисципліни: «Програмування та АМ»

|  |
| --- |
| на тему: «Міжпланетна подорож» |
|  |
|  |

|  |
| --- |
| Виконав: здобувач освіти 2 курсу, групи \_\_\_\_\_\_\_ спеціалізації «Розробка програмного забезпечення»  Пишнов А.Д.  24  \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ (прізвище та ініціали)  Олійник В.В.  Керівник\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_  (підпис) (прізвище та ініціали) |
|  |
|  |
|  |

Полтава – 2024

**ЗМІСТ**

[ВСТУП 3](#_Toc171201472)

[1. ПОСТАНОВКА ЗАДАЧІ 4](#_Toc171201473)

[1.1. Основні вимоги до продукту 5](#_Toc171201474)

[1.2. Вимоги до інтерфейсу 7](#_Toc171201475)

[2. ПЛАНУВАННЯ СИСТЕМИ 9](#_Toc171201476)

[2.1. Алгоритм програмування 9](#_Toc171201477)

[2.2. Тестування 11](#_Toc171201478)

[2.3. Інструкція з використанням системи 12](#_Toc171201479)

[ВИСНОВКИ 13](#_Toc171201480)

[СПИСОК ВИКОРИСТАНОЇ ЛІТЕРАТУРИ 15](#_Toc171201481)

[ДОДАТОК А. ЛІСТИНГ ПРОГРАМИ 16](#_Toc171201482)

# ВСТУП

В сучасному світі інформаційних технологій та комп'ютерних наук, навички програмування та розуміння алгоритмів є фундаментальними для успішного вирішення багатьох задач. Під час виконання цієї курсової роботи, я вдосконалив свої знання та навички з ряду дисциплін, таких як теорія графів, алгоритми та структури даних, а також навички програмування на мові C++. Зокрема, я поглибив своє розуміння алгоритмів пошуку та обробки графових структур, включаючи алгоритм пошуку в ширину.

Мета даної курсової роботи полягає в розробці та тестуванні алгоритму пошуку та знаходження найкоротшого шляху в графі, що представляє собою мережу компаній та маршрутів між ними. Предметом дослідження є алгоритм BFS, його реалізація та оптимізація, а об'єктом дослідження – графові структури даних.

Актуальність роботи обумовлена широким застосуванням графів у різних галузях, таких як соціальні мережі, транспортні системи, мережі зв'язку та багато інших. Ефективні алгоритми пошуку шляхів у графах дозволяють вирішувати важливі практичні задачі, зокрема задачі маршрутизації, планування та оптимізації ресурсів.

# 1. ПОСТАНОВКА ЗАДАЧІ

Для виконання курсової роботи було отримано завдання розробити програмне забезпечення для пошуку найкоротшого шляху в графі за допомогою алгоритму пошуку в ширину (Breadth-First Search, BFS).

Основною метою даного завдання було продемонструвати практичне застосування алгоритму BFS та оцінити його ефективність на прикладі мережі компаній і маршрутів між ними.

Програмний продукт повинен відповідати вимогам. Коректно знаходити найкоротший шлях у графі між заданими вершинами. Алгоритм повинен бути оптимізований для роботи з великими графами, щоб забезпечити швидке виконання. Програма повинна підтримувати роботу з графами різних розмірів, включаючи великі та складні структури. Інтерфейс програми повинен бути простим і зрозумілим для користувача. Взаємодія з програмою здійснюється через командний рядок. Програма повинна бути протестована на різних наборах даних для перевірки її коректності та продуктивності.

Функціональність програми включає:

* **введення даних,** програма приймає на вхід кількість компаній та список маршрутів між ними;
* **графа, на** основі введених даних будується граф, де вершинами є компанії, а ребрами - маршрути між ними;
* **пошук найкоротшого шляху,** використовуючи алгоритм BFS, програма знаходить найкоротший шлях від стартової вершини (першої компанії) до кінцевої вершини (останньої компанії);
* **виведення результату**, кількість кроків у найкоротшому шляху або повідомляє, що до кінцевої вершини неможливо дістатися.
* Для розробки програмного забезпечення було використано програмування C++, інтегроване середовище розробки **(IDE),**  Microsoft Visual Studio та Code::Blocks.

Основними вимогами для реалізації нових функцій є довготривала підтримки та можливості розширення функціональності програми. Код програми повинен бути детально задокументований, що дозволить іншим розробникам швидко розібратися в його структурі та логікі. Програма повинна бути побудована з використанням модульного підходу, що полегшить внесення змін та додавання нових функцій.

**Під час тестування** програма повинна містити набір тестів для перевірки основних функцій та забезпечення їхньої коректності при майбутніх змінах.

**Також** повинна легко адаптуватися до змін у вимогах та можливостей розширення її функціональності за рахунок додавання нових алгоритмів або оптимізації існуючих.

## **Основні вимоги до продукту**

Функціональні вимоги:

* **введення даних,** програма повинна дозволяти користувачу вводити кількість компаній (вершин графа) і кількість маршрутів (ребер графа) також програма повинна дозволяти користувачу вводити конкретні маршрути між компаніями, вказуючи пару компаній для кожного маршруту;
* **побудова графа, п**рограма повинна будувати граф на основі введених даних, де вершинами є компанії, а ребрами – маршрути між ними;
* **пошук найкоротшого шляху п**рограма повинна використовувати алгоритм пошуку в ширину (BFS) для знаходження найкоротшого шляху від першої компанії до останньої, також повинна виводити кількість кроків у найкоротшому шляху або повідомляти, що до кінцевої вершини неможливо дістатися;
* **виведення результату** відображати результат у зрозумілому вигляді, вказуючи кількість кроків у найкоротшому шляху або відповідне повідомлення про неможливість досягнення цілі.

Нефункціональні вимоги:

* **безпека** повинна бути захищена від введення некоректних даних (наприклад, негативних чисел або некоректних пар компаній), не повинна містити вразливостей, які можуть бути використані для порушення її роботи або отримання несанкціонованого доступу до даних;
* **зручність використання** інтерфейсу повинен бути простим і інтуїтивно зрозумілим. Програма повинна надавати користувачу чіткі інструкції щодо введення даних і отримання результатів;
* **масштабованість** підтримує роботу з великими графами, що містять значну кількість вершин і ребер, також ефективно використовувати пам'ять і обчислювальні ресурси для забезпечення швидкого виконання навіть на великих наборах даних;
* **відмовостійкість** - стійкою до помилок і збоїв у разі виникнення помилки, програма повинна надавати користувачу зрозуміле повідомлення про помилку і можливість повторного введення даних для пошуку найкоротшого шляху в графі за допомогою алгоритму BFS.

**Підсистема введення даних в**ідповідає за отримання вхідних даних від користувача (кількість компаній, кількість маршрутів і самі маршрути) забезпечує валідацію введених даних. П**обудови графа в**ідповідає за перетворення введених даних у граф, представлений у вигляді списку суміжності. Створює структуру даних для зберігання графа.

**Підсистема пошуку найкоротшого шляху** пошуку в ширину (BFS) для знаходження найкоротшого шляху в графі, використовує мінімальну кількість переходів між заданими вершинами. **Підсистема виведення результату в**ідповідає за відображення результатів роботи програми користувачу (кількість кроків у найкоротшому шляху або повідомлення про неможливість досягнення цілі).

Програмний продукт побудований з використанням модульного підходу, що забезпечує його гнучкість та можливість легкого розширення в майбутньому. Кожна підсистема реалізована окремо, що полегшує тестування та обслуговування коду.

Використані технології: C++, Microsoft Visual Studio, Code::Blocks, стандартна бібліотека C++ (STL), зокрема вектори та черги. Фреймворки не використовувались.

Програма не має зовнішніх залежностей і не потребує встановлення додаткових компонентів. Все необхідне для її роботи включено до стандартної бібліотеки C++.

Функціональність програми:

* **приймає на вхід кількість компаній і маршрути між ними;**
* **будує граф на основі введених даних;**
* **виконує пошук найкоротшого шляху в графі за допомогою алгоритму BFS;**
* **відображає кількість кроків у найкоротшому шляху або повідомляє про неможливість досягнення цілі.**

Перелік реалізованих за час практики функціональних вимог:

* **введення даних,** функціонал для введення кількості компаній та маршрутів;
* **побудова графа,** на основі введених даних;
* **пошук найкоротшого шляху,** алгоритм BFS для пошуку найкоротшого шляху;
* **виведення результату** пошуку користувачу.

Перелік нереалізованої функціональності: на даний момент всі заплановані функціональні вимоги були реалізовані. Якщо б якісь функції не були реалізовані, причини могли б включати обмеження часу, або потреба у додаткових дослідженнях, непередбачені технічні труднощі.

## **Вимоги до інтерфейсу**

Основні вимоги до інтерфейсу:

* **інтерфейс** має бути простим у використанні, зрозумілим навіть для користувачів з базовими знаннями комп'ютерів;
* **мінімалістичний дизайн,** уникнення надмірного використання графічних елементів, щоб не перевантажувати користувача;
* **зручне розташування елементів** основні функції та команди мають бути легко доступні та розташовані логічно;
* **підтримка кирилиці** всі текстові елементи інтерфейсу повинні підтримувати українську мову та правильно відображати кириличні символи;
* **візуальна відповідність** інтерфейсу має бути візуально приємною і відповідати сучасним стандартам дизайну.

Для створення продукту був використаний фреймворк **Qt**, який забезпечує потужний набір інструментів для створення крос-платформних GUI-додатків. Які в свою чергу були використані для покращення зручності користування та загального користувацького досвіду.

Автоматичне фокусування на основних елементах при запуску програми курсор автоматично фокусується на першому текстовому полі для введення даних.

**Зворотній зв’язок для користувача** візуальні індикатори та повідомлення про помилки або успішні операції забезпечують зрозумілий зворотний зв'язок. Інтерфейс повністю локалізований українською мовою, що підвищує зручність використання для місцевих користувачів. Користувач може обрати темну або світлу тему в залежності від своїх уподобань та умов освітлення.

# 2. ПЛАНУВАННЯ СИСТЕМИ

# 2.1. Алгоритм програмування

Алгоритм роботи:

* користувач вводить кількість планет (n) та кількість компаній (t), також інформацію про кількість планет, що обслуговуються кожною компанією, та номери цих планет;
* на основі введених даних формується граф, де кожна планета є вершиною, а ребра з'єднують планети, які обслуговуються однією компанією;
* виконується алгоритм пошуку в ширину для визначення мінімальної кількості переміщень з планети 0 до планети n-1;
* якщо знайдено шлях, виводиться мінімальна кількість переміщень, якщо шлях не знайдено, виводиться повідомлення про неможливість досягнення цільової планети.

Алгоритм роботи програми представлений на Рисунку 2.1.

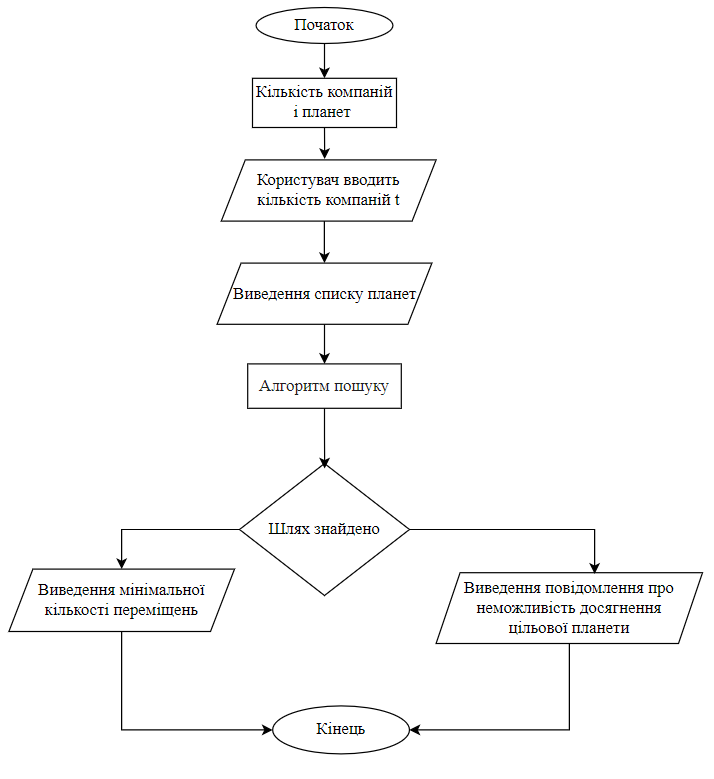
****

Рисунок 2.1 – Алгоритм роботи програми

## **2.2. Тестування**

**Методи тестування програми:**

* модульне тестування функції bfs тестування на простих графах для перевірки базової функціональності, на графах із циклом для перевірки стійкості алгоритму до зациклення та на графах без шляху від стартової до кінцевої планети;
* інтеграційне тестування взаємодії функції bfs з функцією формування графу, виявлена помилка неправильна обробка номерів планет (введені планети починалися з 1 замість 0). Виправлено шляхом зменшення введених значень на 1;
* системне тестування на різних наборах даних маленькі графи (2-5 планет), середні графи (10-50 планет), великі графи (100 і більше планет). Виявлені помилки, переповнення черги на великих графах. Виправлено оптимізацією роботи з чергою. Неправильне виведення результату при відсутності шляху. Виправлено додаванням перевірки на значення -1;
* тестування користувацького інтерфейсу локалізації, перевірка правильності відображення кириличних символів, перевірка на інтуїтивність розташування елементів інтерфейсу. Виявлені помилки неправильне відображення повідомлень про помилки. Виправлено шляхом коригування повідомлень;
* регресійне тестуванняповторне після внесення змін. Перевірка на стабільність програми.

**Зміни внесені до проекту тестування:**

* оптимізація роботи з чергою на більш оптимальну реалізацію для великих графів;
* корекція обробки номерів планет виправлення введених значень планет (зменшення на 1 для переходу від індексації з 1 до індексації з 0);
* поліпшення користувацького інтерфейсу**,** додано додаткові повідомлення про помилки. Оптимізовано відображення кириличних символів;
* виправлення виведення результату**,** додано перевірку на значення -1 для випадків, коли шлях не знайдено.

## **2.3. Інструкція з використанням системи**

Встановлення і запуск програми:

* завантаження програмискачайте архів з програмою з посилання;
* розпакування архівувідкрийте завантажений архів і розпакуйте його у зручну для вас папку;
* компиляція програмивідкриває термінал або командний рядок, у директорію, куди було розпаковано архів;
* запуск програми на екрані з’явиться вітальне повідомлення.

Ця програма розраховує мінімальну кількість зміщень між планетами. Зазначена кількість планет (n) та кількість компаній (t):

Введення кількості планет і компаній кількість планет n і кількість компаній t, після чого натисніть Enter.

Програма попросить вас ввести кількість планет, що обслуговуються кожною компанією, та номери цих планет. Вводимокількість планет, що обслуговуються першою компанією, та номери цих планет, після чого натисніть Enter.

Введення даних про другу компанію кількість планет, що обслуговуються другою компанією, та номери цих планет, після чого натисніть Enter.

Отримання результату програма виконає розрахунок і виведе результат. Якщо шлях знайдено, ви побачите повідомлення про мінімальну кількість переміщень. Якщо шлях не знайдено, ви побачите повідомлення про неможливість досягнення цільової планети.

# ВИСНОВКИ

Під час роботи з курсовою роботою, переді мною було поставлено завдання розробити програмне забезпечення для обчислення мінімальної кількості переміщень між планетами, обслуговуваними різними космічними компаніями. Мета проекту полягала у створенні інструменту, який дозволяє користувачам швидко знаходити оптимальний шлях між планетами, використовуючи алгоритм пошуку в ширину (BFS).

В ході роботи над проектом було:

* розроблено програму на мові програмування C++ з використанням алгоритму BFS для обчислення мінімальної кількості переміщень між планетами;
* реалізовано текстовий інтерфейс для користувача з підтримкою кириличних символів;
* проведено модульне, інтеграційне та системне тестування програми, що дозволило виявити та виправити численні помилки, а також оптимізувати роботу з великими наборами даних;
* підготовлено детальну інструкцію з використання програми для кінцевих користувачів.

Результатом роботи є консольний додаток, який дозволяє вводити дані про кількість планет та компаній, що їх обслуговують, після чого обчислює мінімальну кількість переміщень між планетами. Програма може бути застосована в задачах планування маршрутів у космічних системах, а також для освітніх цілей з вивчення алгоритмів пошуку.

Приклади застосування:

* космічні система, оптимізація маршрутів між планетами для зниження витрат ресурсів;
* освіта, використання програми як навчального інструменту для демонстрації роботи алгоритмів пошуку.

Отримані знання та навички під час роботи над проектом я значно вдосконалив свої знання з:

* алгоритмів та структур даних, зокрема алгоритмів пошуку в графа;
* мови програмування C++ та роботи з текстовими інтерфейсами;
* методи тестування програмного забезпечення, включаючи модульне, інтеграційне та системне тестування;

В подальшому я планую продовжити роботу над проектом, зокрема:

* реалізувати графічний інтерфейс користувача для поліпшення зручності використання програми;
* оптимізувати програму для роботи з дуже великими наборами даних;
* розширити функціонал програми, додавши можливість збереження та завантаження даних.

Знання, отримані під час вивчення дисциплін «Алгоритми та структури даних», «Програмування на C++», «Тестування програмного забезпечення», значно сприяли успішній реалізації цього проекту. Створений мною продукт є важливою частиною моєї курсової роботи.

Під час виконання цієї курсової роботи я отримав цінний досвід у розробці та тестуванні алгоритмів, що дозволило вдосконалити мої програмістські навички та поглибити розуміння теоретичних основ комп'ютерних наук. Це, у свою чергу, сприяє підготовці до подальших професійних викликів та досліджень у галузі інформаційних технологій.

# СПИСОК ВИКОРИСТАНОЇ ЛІТЕРАТУРИ

1. Васильєв О. Програмування С++ в прикладах і задачах. 2-ге вид. Київ : Ліра-К, 2017. 480 с.
2. Скотт Мейерс. Ефективне сучасне програмування на C++: 42 специфічні способи покращити використання C++11 і C++14. . Київ: Пер. з англійсько, 2014 – 334 с.
3. Б'ярне Страуструп. Шлях до C++: теорія та практика з використанням C++11. Київ: Пер. з англ., 2013 – 220.
4. Основи програмування на С++ для починаючих. [Електронний ресурс] – Режим доступу до ресурсу: <https://purecodecpp.com/>
5. МЕТОДИЧНІ ВКАЗІВКИ ДО ВИКОННАННЯ КУРСОВИХ РОБІТ ДЛЯ ЗДОБУВАЧІВ ОСВІТИ СПЕЦІАЛЬНОСТІ 121 «ІНЖЕНЕРІЯ ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ»: МЕТОДИЧНІ ВКАЗІВКИ / Відокремлений структурний підрозділ «Полтавський політехнічний фаховий коледж Національного технічного університету «Харківський політехнічний інститут»; [уклад.: В.В.Олійник]. – Полтава: ВСП ППФК НТУ “ХПІ”, 2022. – 22 с.

# ДОДАТОК А. Лістинг програми

