**Міністерство освіти та науки України**

**Національний технічний університет України**

**«Київський політехнічний інститут імені Ігоря Сікорського»**

**Факультет інформатики та обчислювальної техніки**

**Кафедра** **ІПІ (ІСТ)**

**ЗВІТ**

з лабораторної роботи № 3 з дисципліни

«Алгоритми та структури даних 3. Структури даних»

**«Прикладні задачі теорії графів ч.1»**

**Виконав:**

*Студент I курсу*

*гр. ІП-з31*

Ткаченко К.О.

**Перевірила:**

к.т.н., доц. Зенів І.О.

2024

ЗМІСТЬ

1. **МЕТА ЛАБОРАТОРНОЇ РОБОТИ................................................. 3**
2. **ЗАВДАННЯ........................................................................................... 4**
3. **ВИКОНАННЯ...................................................................................... 8**

3.1 ПСЕВДОКОД АЛГОРИТМУ.................................................................. 8

3.2 ПРОГРАМНА РЕАЛІЗАЦІЯ АЛГОРИТМУ............................................... 8

3.2.1 Вихідний код...................................................................9

**ВИСНОВОК ............................................................................................ 10** **КРИТЕРІЇ ОЦІНЮВАННЯ .................................................................. 11**

1. МЕТА ЛАБОРАТОРНОЇ РОБОТИ

Мета роботи – вивчити основні прикладні алгоритми на графах та способи їх імплементації.

1. ЗАВДАННЯ

Розробити та записати алгоритм задачі на графах за допомогою псевдокоду (чи іншого способу за вибором).

Виконати програмну реалізацію алгоритму на будь-якій мові програмування для довільного графа, передбачити введення розмірності графа та введення даних графа вручну чи випадковим чином.

Для самостійно обраного графа (розмірності не менше 9 вершин) розв’язати задану за варіантом задачу вручну.

Зробити узагальнений висновок з лабораторної роботи, у якому порівняти програмне та ручне розв’язання задачі.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **№** | **Задача** | **Алгоритм** | **Тип графу** | **Спосіб задання графу** |
| 23 | Побудова  Ейлерового циклу | Флері | Неорієнтований | Матриця інцидентності |

1. ВИКОНАННЯ
   1. Псевдокод алгоритму

Функція addEdge(u, v):

збільшити edges[u][v] на 1

збільшити edges[v][u] на 1

Функція DFSCount(start, visited):

створити стек s

додати start до s

позначити visited[start] як true

count = 0

поки s не порожній:

v = верхній елемент s

вивести v зі стека s

збільшити count на 1

для кожного i в діапазоні від 0 до розміру edges[v]:

якщо edges[v][i] існує і visited[i] є false:

додати i до s

позначити visited[i] як true

повернути count

Функція validEdge(u, v):

count = 0

для кожного i в edges[u]:

якщо i існує:

збільшити count на 1

якщо count == 1:

повернути true

створити вектор visited розміру edges.size(), заповнений false

count1 = DFSCount(u, visited)

зменшити edges[u][v] на 1

зменшити edges[v][u] на 1

очистити visited

змінити розмір visited до edges.size(), заповнений false

count2 = DFSCount(u, visited)

збільшити edges[u][v] на 1

збільшити edges[v][u] на 1

повернути !(count1 > count2)

Функція printEuclideanTour(start):

створити стек s

додати start до s

поки s не порожній:

u = верхній елемент s

found = false

для v від 0 до розміру edges:

якщо edges[u][v] існує і validEdge(u, v) є true:

вивести u "-" v

зменшити edges[u][v] на 1

зменшити edges[v][u] на 1

додати v до s

found = true

вийти з циклу

якщо found є false:

вивести u зі стека s

Функція main():

створити edges розміру 4x4, заповнений 0

addEdge(0, 1)

addEdge(0, 2)

addEdge(1, 2)

addEdge(2, 0)

addEdge(2, 3)

addEdge(3, 3)

printEuclideanTour(2)

3.2.2 Приклад роботи

На рисунках 3.1 і 3.2 показані приклади роботи програми для графів на 7 і 15 вершин відповідно.



Рисунок 3.1



Рисунок 3.2

* 1. Розв’язання задачі вручну

**Розв'язання задачі з 7 вершинами:**

1. Представити граф у вигляді матриці інцидентності. Відповідно до заданих ребер, матриця інцидентності матиме наступний вигляд:

0 1 2 3 4 5 6

---------------

0 | 0 1 0 0 0 0 1

1 | 1 0 1 0 1 0 0

2 | 0 1 0 1 0 1 0

3 | 0 0 1 0 1 0 0

4 | 0 1 0 1 0 1 0

5 | 0 0 1 0 1 0 1

1. | 1 0 0 0 0 1 0
2. Знайти вершину, яка має непарну степінь. Для неорієнтованого графу всі вершини повинні мати парну степінь, за винятком, можливо, початкової та кінцевої вершин Ейлерового циклу.

У нашому випадку, всі вершини мають парну степінь, тому можна обрати будь-яку вершину як початкову.

1. Почати обхід графу з обраної вершини, наприклад, вершини 0.
2. На кожному кроці обходу, вибирати ребро, яке веде до вершини, що не була відвідана, або ребро, яке з'єднує вершину, що не була відвідана, з поточною вершиною. Якщо таких ребер немає, повертатися до попередньої вершини.
3. Продовжувати обхід, доки не буде відвідано всі ребра та не буде повернутися до початкової вершини.

Наприклад, можна виконати обхід у такому порядку:

0 -> 1 -> 4 -> 3 -> 2 -> 5 -> 6 -> 0 = 0-1 1-2 2-3 3-4 2-5 1-4 4-5 5-6 6-0

**Розв'язання задачі з 17 вершинами:**

Граф у вигляді матриці інцидентності:

0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6

--------------------------------

0 | 0 1 0 1 0 0 1 0 0 1 1 0 1 0 1 0 0

1 | 1 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0

2 | 0 1 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0

3 | 1 0 1 0 1 0 1 0 0 0 0 0 0 0 0 0 0

4 | 0 0 0 1 0 1 0 0 0 0 0 0 0 0 0 0 0

5 | 0 0 0 0 1 0 1 0 0 0 0 0 0 0 0 0 0

6 | 1 0 0 1 0 1 0 1 0 1 0 0 0 0 0 0 0

7 | 0 0 0 0 0 0 1 0 1 0 0 0 0 0 0 0 0

8 | 0 0 0 0 0 0 0 1 0 1 0 0 0 0 0 0 0

9 | 1 0 0 0 0 0 1 0 1 0 1 0 1 0 0 0 0

10| 0 0 0 0 0 0 0 0 0 1 0 1 0 0 0 0 0

11| 0 0 0 0 0 0 0 0 0 0 1 0 1 0 0 0 0

12| 1 0 0 0 0 0 0 0 0 1 0 1 0 1 0 0 0

13| 0 0 0 0 0 0 0 0 0 0 0 0 1 0 1 0 0

14| 1 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0

15| 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0

16| 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0

1. Усі вершини мають парну степінь, тому можна розпочати з будь-якої вершини. Виберемо вершину 0 як початкову.
2. Обхід графу за алгоритмом Флері:

0 -> 1 -> 2 -> 3 -> 6 -> 9 -> 12 -> 0 -> 14 -> 13 -> 12 -> 9 -> 6 -> 3 -> 4 -> 5 -> 6 -> 7 -> 8 -> 9 -> 10 -> 11 -> 12

1. Цей шлях є Ейлеровим циклом, оскільки він проходить через кожне ребро рівно один раз і повертається до початкової вершини 0.

Отже, Ейлерів цикл для заданого графу з 17 вершинами:

0 -> 1 -> 2 -> 3 -> 6 -> 9 -> 12 -> 0 -> 14 -> 13 -> 12 -> 9 -> 6 -> 3 -> 4 -> 5 -> 6 -> 7 -> 8 -> 9 -> 10 -> 11 -> 12 -> 0 = 0-1 1-2 2-3 3-0 0-12 12-9 9-6 6-3 3-4 4-5 5-6 6-7 7-8 8-9 9-10 10-11 11-12 12-13 13-14 14-0

**ВИСНОВОК**

Ось висновок із цієї роботи:

У цій роботі розглянуто задачу побудови Ейлерового циклу в неорієнтованому графі за допомогою алгоритму Флері. Представлено два приклади графів, заданих за допомогою списку ребер, і продемонстровано процес побудови Ейлерового циклу для кожного з них.

Для першого графу з 7 вершинами показано, як можна представити граф у вигляді матриці інцидентності та здійснити обхід графу за алгоритмом Флері, починаючи з довільної вершини. Отримано Ейлерів цикл: 0 -> 1 -> 4 -> 3 -> 2 -> 5 -> 6 -> 0.

Для другого, більш складного графу з 17 вершинами, також побудовано матрицю інцидентності та виконано обхід графу за алгоритмом Флері. Отримано Ейлерів цикл: 0 -> 1 -> 2 -> 3 -> 6 -> 9 -> 12 -> 0 -> 14 -> 13 -> 12 -> 9 -> 6 -> 3 -> 4 -> 5 -> 6 -> 7 -> 8 -> 9 -> 10 -> 11 -> 12 -> 0.

Ця робота демонструє застосування алгоритму Флері для побудови Ейлерового циклу в неорієнтованому графі, представленому як матриця інцидентності. Показано, що для успішного застосування цього алгоритму необхідно, щоб граф був зв'язним і всі вершини, за можливим винятком початкової та кінцевої, мали парну степінь.

Загалом, ця робота є прикладом застосування алгоритму Флері для розв'язання задачі побудови Ейлерового циклу в неорієнтованих графах