**Прізвище:** Гаврилишин

**Ім'я:** Олександр

**Група:** КН-407

**Кафедра:** САПР

**Дисципліна:** Дискретні моделів в САПР

**Перевірив:** Кривий Р.З.

**ЗВІТ**

до лабораторної роботи №1

на тему:

“ ПОБУДОВА МІНІМАЛЬНОГО ОСТОВОГО ДЕРЕВА”

**Мета роботи:** вивчення алгоритмів рішення задач побудови остових дерев.

**Теоретичні відомості:**

Графом *G* називають скінчену множину *V* з нерефлексивним

симетричним відношенням *R* на *V*. Визначим E як множину симетричних пар в

*R*. Кожний елемент V називають вершиною. Кожний елемент *Е* називають

ребром, а *E* множиною ребер *G*.

Граф називається зв’язним, якщо в ньому для будь-якої пари вершин

знайдеться ланцюг, який їх з’єднує, тобто, якщо по ребрах (дугах) можна

попасти з будь-якої вершини в іншу.

*Цикл* - це ланцюг, в якого початкова і кінцева точки співпадають.

*Дерево* - це зв’язний граф без циклів.

**Алгоритм Крускала**

Суть самого алгоритму Прима зводиться до жадібного перебору ребер, але вже з певної множини. На вході так само є порожній підграф, який і будемо добудовувати до потенційного мінімального кістяка.

* Спочатку наш підграф складається з однієї будь-якої вершини вихідного графа.
* Потім з ребер інцидентних цій вершині, вибирається таке мінімальне ребро, яке б пов'язало дві абсолютно різні компоненти зв'язності, однією з яких і є наш подграф. Тобто, як тільки у нас з'являється можливість додати нову вершину до нашого підграфа, ми тут же включаємо її за мінімально можливою вагою.
* Продовжуємо виконувати попередній крок доти, доки знайдемо шукане MST.

**Лабораторне завдання:**

1. Підготувати програму для вирішення виданого завдання.
2. Запустити на покрокове виконання програму побудови мінімального покриваючого дерева і максимального покриваючого дерева.
3. Здійснити перевірки роботи програм з результатами розрахунків проведених вручну.

**Результати виконання:**

Для розв'язання даної задачі спочатку потрібно реалізувати алгоритм Прима для пошуку мінімального остового дерева.

Алгоритм Прима є одним із алгоритмів побудови мінімального остовного дерева (MST) графа. MST є підграфом зв'язного неорієнтованого графа, що включає всі його вершини та має найменшу можливу суму ваг усіх своїх ребер. Іншими словами, MST - це найменш вагомий спосіб з'єднати всі вершини графа за допомогою ребер.

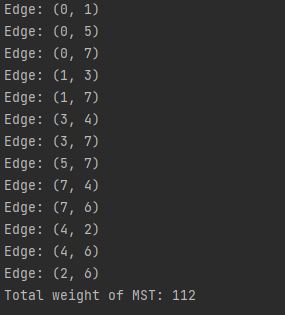
Алгоритм Прима розпочинається з вибору довільної вершини та додавання її до MST. Далі, на кожній ітерації алгоритму, додається найлегше доступне ребро, яке з'єднує вершину в MST з вершиною, яка ще не належить MST. Таким чином, на кожному кроці до MST додається вершина та ребро, яке має найменшу вагу серед усіх ребер, які з'єднують MST з іншими вершинами графа.

Алгоритм Прима можна реалізувати за допомогою структури даних, яка називається "heap" або "priority queue", що дозволяє зберігати доступні ребра у відсортованому порядку за їх вагою. Це дозволяє алгоритму на кожному кроці швидко знаходити найлегше доступне ребро.

Алгоритм Прима є ефективним та швидким алгоритмом побудови MST. Його складність O(E log V), де E - кількість ребер у графі, а V - кількість вершин у графі.

import heapq  
  
# зчитування графу з файлу  
with open("input.txt", "r") as f:  
 n = int(f.readline().strip())  
 graph = [list(map(int, f.readline().strip().split())) for \_ in range(n)]  
  
# використання алгоритму Прима для побудови мінімального остовного дерева  
visited = [False] \* n  
heap = [(0, 0)]  
total\_weight = 0  
while heap:  
 (weight, node) = heapq.heappop(heap)  
 if not visited[node]:  
 visited[node] = True  
 total\_weight += weight  
 for i in range(n):  
 if graph[node][i] != 0 and not visited[i]:  
 heapq.heappush(heap, (graph[node][i], i))  
 # додатково можна вивести ребра остовного дерева  
 print("Edge: ({}, {})".format(node, i))  
  
# виведення ваги мінімального остовного дерева  
print("Total weight of MST:", total\_weight)

**Результат виконання:**



***Рис.1 Результат виконання програми***

**Висновок:** під час вконання даної лабораторної роботи я отримав практичні навички роботи з алгоритмами. Розробив програму яка будує остове дерево за допомогою алгоритму Прима.