Національний технічний університет України «КПІ ім. Ігоря Сікорського»

Факультет Інформатики та Обчислювальної Техніки

Кафедра Автоматизованих Систем Обробки Інформації та Управління

Лабораторна робота №3

з дисципліни «Дискретна математика»

на тему

«**Обхід графів**»

Виконав:

студент гр. ІС-91

Косюк Михайло

Викладач:

доц. Рибачук Л.В.

Київ – 2019

### Зміст

[Зміст 2](#_Toc190845267)

[1 Постановка задачі 3](#_Toc190845268)

[2 Результати виконання програми 4](#_Toc190845269)

[3 Лістинг програми](#_Toc190845271) 5

### 1 Постановка задачі

Ціль

Дослідити роботу алгоритмів обходу в графах: пошуку вшир та пошуку вглиб.

Завдання

Реалізувати програмне застосування (програму), яке виконує наступні функції. Причому на вхід програми подається вхідний файл з описом графу, зі структурою, яка вказана у практичному завданні №1 «Представлення графів». *При реалізації алгоритмів вважати, що заданий граф є зв’язаним*.

1. Обійти граф пошуком вшир. Користувач вводить початкову вершину графу. Програма виконує обхід графу, починаючи з вказаної початкової вершини. На екран виводиться протокол обходу – таблиця, яка містить наступні дані по кожній ітерації алгоритму обходу: поточна вершина, її BFS-номер, вміст черги (див. тему 28 електронного конспекту).

2. Обійти граф пошуком углиб. Аналогічно за пунктом 1 завдання, але програма виконує обхід графу пошуком углиб. На екран виводиться протокол обходу: поточна вершина, її DFS-номер, вміст стеку.

Введення даних здійснюється у файл condition.txt, що знаходиться в одній директорії із фалом програми, згідно із заданим форматом.

Тестовий граф, що використовувався для обробки заданий такою послідовністю(граф, що містить 8 вершин і 9 ребер):

**8 9**

**2 4**

**4 1**

**3 7**

**2 3**

**3 8**

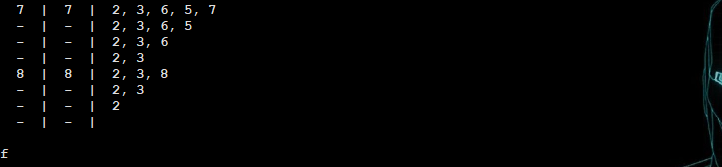
**4 3**

**3 6**

**5 6**

**5 7**

### 2 Результати виконання програми



### 3 Лістинг програми

from itertools import islice  
  
def do\_bfs(edges, tops):  
 start = int(input("Введіть вершину початок: "))  
 if start not in range(1, tops+1):  
 print("У графі не існує заданої вершини")  
 exit()  
 queue = [start]  
 tops = [t for t in range(1, tops+1) if t != start]  
 num = 1  
 for e in edges: e.sort()  
 edges.sort()  
 print(" v |BFS-n| queue content")  
 print("---------------------------")  
 print("%3d |%3d |" % (start, num) + " " + str(queue).strip('[]'))  
 while queue:  
 start = queue[0]  
 for p in edges:  
 if start in p and p[0 if p.index(start) else 1] in tops:  
 i = 0 if p.index(start) else 1  
 queue.append(p[i])  
 tops.remove(p[i])  
 num += 1  
 print("%3d |%3d |" % (p[i], num) + " " + str(queue).strip('[]'))  
 del queue[0]  
 print(" - | - | " + str(queue).strip('[]'))  
  
  
def do\_dfs(edges, tops):  
 start = int(input("Введіть вершинк початок: "))  
 if start not in range(1, tops+1):  
 print("У графі не існує заданої вершини")  
 exit()  
 stack = [start]  
 tops = [t for t in range(1, tops+1) if t != start]  
 num = 1  
 for e in edges: e.sort()  
 edges.sort()  
 print(" v |DFS-n| stack content")  
 print("---------------------------")  
 print("%3d |%3d |" % (start, num) + " " + str(stack).strip('[]'))  
 while stack:  
 start = stack[len(stack)-1]  
 cur\_v = [e[0] if e.index(start) else e[1] for e in edges if start in e and e[0 if e.index(start) else 1] in tops]  
 while cur\_v:  
 cur\_v = [e[0] if e.index(start) else e[1] for e in edges if start in e and e[0 if e.index(start) else 1] in tops]  
 if cur\_v and cur\_v[0] in tops:  
 stack.append(cur\_v[0])  
 tops.remove(cur\_v[0])  
 num += 1  
 start = cur\_v[0]  
 print("%3d |%3d |" % (cur\_v[0], num) + " " + str(stack).strip('[]'))  
 del stack[len(stack) - 1]  
 print(" - | - | " + str(stack).strip('[]'))  
  
  
file = open('condition.txt')  
tops\_num = int(file.read(1))  
edges\_num = int(file.read(3))  
edge\_list = []  
for edge in islice(file, 0, edges\_num):  
 ep = []  
 for top in edge.split(' ', 1):  
 ep.append(int(top))  
 edge\_list.append(ep)  
if edges\_num != len(edge\_list):  
 print("\nКількість ребер не співпадає у із заданим списком, перевірте файл condition.txt")  
 exit()  
print("Кількість вершин: " + str(tops\_num))  
print("Кількість ребер: " + str(edges\_num))  
print("Список граней: ", edge\_list)  
  
print(  
 "\nНатисніть 1 щоб обійти граф вшир BFS"  
 "\nНатисніть 2 щоб обійти граф вглиб DFS"  
 "\nPress f щоб завешити роботу"  
)  
control = ""  
while control != "f":  
 control = input('\n')  
 if control == '1': do\_bfs(edge\_list, tops\_num)  
 if control == '2': do\_dfs(edge\_list, tops\_num)