Міністерство освіти і науки України

Національний технічний університет України

“Київський політехнічний інститут ім. Ігоря Сікорського”

Факультет інформатики та обчислювальної техніки

Кафедра автоматизованих систем обробки інформації та управління

ЗВІТ

про виконання лабораторного практикуму №8(2)

Виконав:

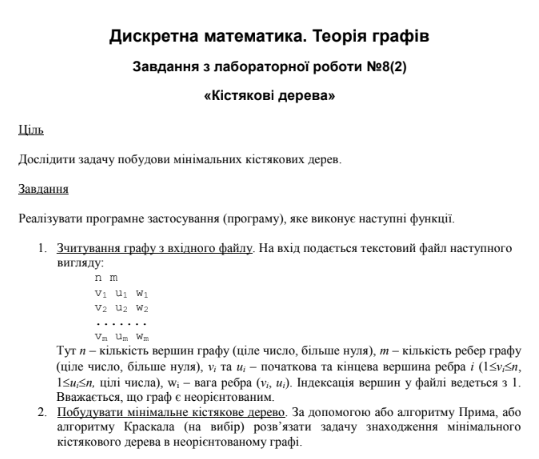
студент 1- го курсу ФІОТ

групи *ІП-91*

*Кінчур Вадим Вікторович*

Київ 2020

1. Умова лабораторної роботи

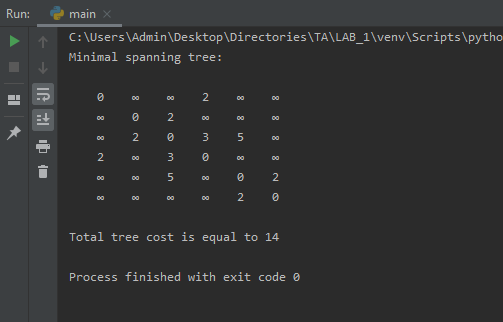


1. Програмний код (Python):

# Constants  
INF = float('inf')  
  
def get\_matrix\_sum(matrix):  
 return sum([val for row in matrix for val in row if val != INF]) // 2  
  
# Print MST  
def print\_min\_tree(adj\_m):  
 to\_print = process\_prim(adj\_m)  
 if to\_print is None:  
 return print("Minimal spanning tree doesn't exist!")  
 print("Minimal spanning tree:\n")  
 for i in range(len(to\_print)):  
 for j in range(len(to\_print)):  
 if i != j:  
 out\_val = to\_print[i][j] if to\_print[i][j] != INF else '\u221e'  
 else:  
 out\_val = 0  
 print(f'{out\_val:>5}', end = '')  
 print()  
 print(f"\nTotal tree cost is equal to {get\_matrix\_sum(to\_print)}")  
  
# Prim algorithm   
def process\_prim(adj\_m):  
 v\_num = len(adj\_m)  
 used = [False] \* v\_num  
 sel\_edge = [-1] \* v\_num  
 min\_edge = [INF] \* v\_num  
 mst = [[INF] \* v\_num for \_ in range(v\_num)]  
  
 min\_edge[0] = 0  
 for i in range(v\_num):  
 v = -1  
 # Find new start vertex  
 for j in range(v\_num):  
 if not used[j] and (v == -1 or min\_edge[j] < min\_edge[v]):  
 v = j  
 # Check is possible to build MST  
 if min\_edge[v] == INF:  
 return None  
 # Build MST  
 used[v] = True  
 if sel\_edge[v] != -1:  
 mst[v][sel\_edge[v]] = min\_edge[v]  
 mst[sel\_edge[v]][v] = min\_edge[v]  
 # Try to find minimal edge from v  
 for to in range(v\_num):  
 if adj\_m[v][to] < min\_edge[to]:  
 min\_edge[to] = adj\_m[v][to]  
 sel\_edge[to] = v  
 return mst  
   
# Parse input file  
def get\_input(file\_name):  
 with open(file\_name, "r") as file\_handler:  
 v\_num, e\_num = map(int, file\_handler.readline().split())  
 adj\_m = [[INF] \* v\_num for \_ in range(v\_num)]  
 for \_ in range(e\_num):  
 start, end, weight = map(int, file\_handler.readline().split())  
 adj\_m[start - 1][end - 1] = weight  
 adj\_m[end - 1][start - 1] = weight  
 return adj\_m  
  
# Main method  
def main():  
 adj\_m = get\_input('input.txt')  
 print\_min\_tree(adj\_m)  
   
main()

1. Результат виконання:

**Вхідні дані:**

6 9

1 2 5

1 5 7

1 4 2

2 3 2

2 4 5

3 4 3

3 5 5

4 6 8

5 6 2