Ministry of Education and Science of Ukraine

National Technical University of Ukraine

«Kyiv Polytechnic Institute. Igor Sikorsky »

Faculty of Informatics and Computer Technologies

Department of Computer Engineering

LAB № 7

from the discipline "Theory of Algorithms"

on the topic «Greedy algorithms»

PERFORMED BY:

1st year student

group ІП-93

Zavalniuk M.E.

The credit - 9312

Variant – 12

CHECKED:

Associate Professor of OT

c.t.s.,s.r.

Antoniuk А.І.

Kiev - 2020

**TASK**

**Goal:**

implementation of the greedy algorithm for the traveling salesman task.

**Option task:**

In this work, it is necessary to propose a greedy algorithm for the task of a salesman.

The task of the salesman is formulated for a complete graph. For a weighted complete graph G with n vertices, the distances between all pairs of vertices (i, j) are given. You need to find the shortest route that runs through all vertices of the graph and enters each vertex only once.

**CODE**

# Import function

**from** sys **import** maxsize

V **=** 4

# implementation of traveling Salesman Problem. Algorithm itself

**def** travellingSalesmanProblem**(**graph**,** s**):**

# store all vertex apart from source vertex

vertex **=** **[]**

**for** i **in** **range(**V**):**

**if** i **!=** s**:**

vertex**.**append**(**i**)**

# store minimum weight Hamiltonian Cycle

min\_path **=** maxsize

**while** **True:**

# store current Path weight(cost)

current\_pathweight **=** 0

# compute current path weight

k **=** s

**for** i **in** **range(len(**vertex**)):**

current\_pathweight **+=** graph**[**k**][**vertex**[**i**]]**

k **=** vertex**[**i**]**

current\_pathweight **+=** graph**[**k**][**s**]**

# update minimum

min\_path **=** **min(**min\_path**,** current\_pathweight**)**

**if** **not** next\_permutation**(**vertex**):**

**break**

**return** min\_path

# next\_permutation implementation

**def** next\_permutation**(**L**):**

n **=** **len(**L**)**

i **=** n **-** 2

**while** i **>=** 0 **and** L**[**i**]** **>=** L**[**i **+** 1**]:**

i **-=** 1

**if** i **==** **-**1**:**

**return** **False**

j **=** i **+** 1

**while** j **<** n **and** L**[**j**]** **>** L**[**i**]:**

j **+=** 1

j **-=** 1

L**[**i**],** L**[**j**]** **=** L**[**j**],** L**[**i**]**

left **=** i **+** 1

right **=** n **-** 1

**while** left **<** right**:**

L**[**left**],** L**[**right**]** **=** L**[**right**],** L**[**left**]**

left **+=** 1

right **-=** 1

**return** **True**

# Driver Code

**if** \_\_name\_\_ **==** "\_\_main\_\_"**:**

# matrix representation of graph. Input data

graph **=** **[[**0**,** 10**,** 15**,** 20**],** **[**10**,** 0**,** 35**,** 25**],**

**[**15**,** 35**,** 0**,** 30**],** **[**20**,** 25**,** 30**,** 0**]]**

s **=** 0

# Output data

**print(**'Minimum weight:'**,**travellingSalesmanProblem**(**graph**,** s**))**

**RESULTS OF THE PROGRAM WORK**

The input array is:

[[0, 10, 15, 20], [10, 0, 35, 25],

[15, 35, 0, 30], [20, 25, 30, 0]]

Output array:

Minimum weight: 80

**CONCLUSIONS**

I got acquainted with the topic of laboratory work.

Have acquired relevant work skills.

An appropriate test program has been developed.

Learn the story of Traveling Salesman Problem. Understand how greedy algorithms work.