**МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ**

**Чорноморський національний університет   
імені Петра Могили**

Факультет комп’ютерних наук

Кафедра «Інтелектуальних інформаційних систем»

**ЗВІТ**

*з лабораторної роботи № 7*

Дисципліна "Теорія алгоритмів"

Напрям підготовки: 122 Комп’ютерні науки

ЛР.ПЗ.07-201.1910109

***Cтудент*** *\_\_\_\_Грабовський Є.О.\_*

*(підпис)*

*\_\_\_\_\_\_\_\_04.12.2020\_\_\_\_\_\_\_\_ \_\_*

*(дата)*

***Викладач*** *\_\_\_Гожий О.П.*

*(підпис)*

*(дата)*

**Практична робота №7**

**Тема: Алгоритм Дейкстри.**

***Завдання на роботу: 1.***Побудувати алгоритм Дейкстри за заданими вхідними даними. Скласти програмний код. Визначити складність алгоритму. Визначити швидкодію алгоритму. Зробить висновки. (19.10.2020). ***2.***Протестувати програмний код алгоритму Дейкстри. Визначити складність алгоритму. Визначити швидкодію алгоритму. Зробить висновки.

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Варіант №** | **Ваги ребер графу** | | | | | | | | | | | | | | | | **Шлях** |
| **1** | **2** | **3** | **4** | **5** | **6** | **7** | **8** | **9** | **10** | **11** | **12** | **13** | **14** | **15** | **16** |
| 8 | 32 | 14 | 14 | 12 | 34 | 43 | 12 | 32 | 23 | 45 | 32 | 23 | 24 | 83 | 5 | 24 | 3-8 |

Реалізація алгоритму на мові Python:

*# Библиотека для максимального значения числа int*

*import* sys

class Graph():

*# Создание графа с определенным кол-во вершин*

    def \_\_init\_\_(*self*, *vertices*):

        self.V = vertices

        self.graph = [[0 *for* column *in* range(vertices)]

*for* row *in* range(vertices)]

*# Вывод всех вершин и расстояние до них от выбраной вершины*

    def printSolution(*self*, *dist*):

        print("Вершина : расстояние до выбраной вершины")

*for* node *in* range(self.V):

            print(node+1, ':', dist[node])

*# Метод для обработки минимального расстояния до вершины*

    def minDistance(*self*, *dist*, *sptSet*):

*# Минимальное расстояние для следующей вершины*

        min = sys.maxsize

*# Search not nearest vertex not in the*

*# shortest path tree*

*for* v *in* range(self.V):

*if* dist[v] < min and sptSet[v] == False:

                min = dist[v]

                min\_index = v

*return* min\_index

*# Реализация алгоритма Дейкстры*

    def dijkstra(*self*, *src*):

        src -= 1

        dist = [sys.maxsize] \* self.V

        dist[src] = 0

        sptSet = [False] \* self.V

*for* \_ *in* range(self.V):

*# Выбираем минимальную дистанцию до вершины с множеств вершин, которые не были помеченными*

            u = self.minDistance(dist, sptSet)

*# Добавляем минимальную дистанцию до вешины в дерево коротких путей*

            sptSet[u] = True

*# Обновляем расстояние выбраных соседних вершин*

*# если текущее растояние больше нового и*

*# если вершина не находиться в дереве коротких путей*

*for* v *in* range(self.V):

*if* self.graph[u][v] > 0 and \

                        sptSet[v] == False and \

                        dist[v] > dist[u] + self.graph[u][v]:

                    dist[v] = dist[u] + self.graph[u][v]

        self.printSolution(dist)

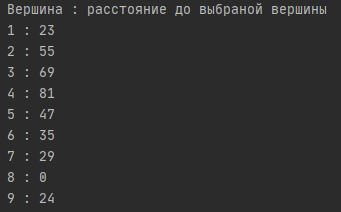
g = Graph(9)

*#   1  2   3  4  5  6   7  8   9*

g.graph = G = [  
 [0, 32, 0, 0, 32, 12, 0, 23, 0],  
 [32, 0, 14, 0, 12, 43, 0, 0, 0],  
 [0, 14, 0, 14, 32, 0, 0, 0, 0],  
 [0, 0, 14, 0, 34, 0, 83, 0, 0],  
 [0, 12, 32, 34, 0, 12, 24, 0, 0],  
 [32, 43, 0, 0, 12, 0, 23, 45, 32],  
 [0, 0, 0, 0, 24, 23, 0, 0, 5],  
 [23, 0, 0, 0, 0, 45, 0, 0, 24],  
 [0, 0, 0, 0, 0, 32, 5, 24, 0]  
]

g.dijkstra(8)

**Результат виконання програми:**



**Визначення швидкодії / складності алгоритмів**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Тип сортування | Кращий час | Середній час | Гірший час | Складність сортування |
| Швидке сорутвання | O(n^2+m) | O(n \log n + m) | O(\log n) | O(n logn) |

**Висновок:** в даній лабораторній роботі ми використали алгоритм Дейкстри. Ми зробили модель-приклад графа у вигляді матриці та знайшли найкоротший шлях від двох заданих вершин.

Взагалі, алгоритм Дейкстри знаходить оптимальні маршрути та їх довжину між двома вершинами. Недолік цього алгоритму полягає у тому, що він буде некоректно працювати, якщо граф має від’ємні значення дуг.