Міністерство освіти і науки України

Харківський національний університет радіоелектроніки

Кафедра інформаційних управляючих систем

Дисципліна: «Дискретна математика»

**ЛАБОРАТОРНА РОБОТА № 4**

**«НАЙКОРОТШІ ШЛЯХИ. АЛГОРИТМ ДЕЙКСТРИ»**

|  |  |  |
| --- | --- | --- |
| Виконав:  ст. гр. ІТУ-19-2  Куренков Богдан Михайлович |  | Прийняла:  Васильцова Н.В.  з оцінкою «\_\_»  «\_\_\_\_»\_\_\_\_\_\_\_\_\_\_\_2020р. |
|  |  |  |

Харків 2020

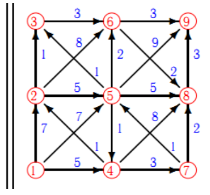
**МЕТА РОБОТИ**

Ознайомлення з основними методами та алгоритмами розв’язання задач відшукання оптимальних шляхів в орієнтованих графах (алгоритмами Дейкстри, Форда, Флойда, Данцига). Вивчення формального опису алгоритму Дейкстри побудови найкоротших шляхів із вершини графа. Програмна реалізація алгоритму Дейкстри для знаходження відстані між парами вершин в орієнтованих і неорієнтованих графах.

**ПОСТАНОВКА ЗАДАЧІ**

**Розробити програму, яка визначає мінімальну відстань від заданої вершини до всіх інших вершин графа.**

**Варіант 9**



**ХІД РОБОТИ**

#include <iostream>

#include <iomanip>

int\* Dijkstra(int\*\* graph, int dimension, int edge);

int main() {

int\*\* graph, dimension = 9;

graph = new int\* [dimension];

for (int i = 0; i < dimension; i++)

graph[i] = new int[dimension];

int data[] = {

0, 7, 0, 5, 7, 0, 0, 0, 0,

0, 0, 1, 0, 5, 8, 0, 0, 0,

0, 0, 0, 0, 0, 3, 0, 0, 0,

0, 1, 0, 0, 0, 0, 3, 8, 0,

0, 0, 1, 1, 0, 2, 0, 5, 9,

0, 0, 0, 0, 0, 0, 0, 2, 3,

0, 0, 0, 0, 0, 1, 0, 2, 0,

0, 0, 0, 0, 0, 0, 0, 0, 3,

0, 0, 0, 0, 0, 0, 0, 0, 0

};

for (int i = 0, k = 0; i < dimension; ++i)

for (int j = 0; j < dimension; ++j, ++k) graph[i][j] = data[k];

int\*\* distance = new int\*[dimension];

for (int i = 0; i < dimension; ++i)

distance[i] = Dijkstra(graph, dimension, i);

for (int i = 0; i < dimension; i++, std::cout << "\n") {

std::cout << "Distance beetween the point x" << i + 1 << " and the others:\n\n";

for (int j = 0; j < dimension; j++)

if (distance[i][j] != INT\_MAX)

std::cout << "x" << i + 1 << " -> x" << j + 1 << " = " << distance[i][j] << "\n";

else std::cout << "x" << i + 1 << " -> x" << j + 1 << " = inf\n";

}

return 0;

}

int\* Dijkstra(int\*\* graph, int dimension, int edge) {

int\* distance = new int[dimension];

bool\* visited = new bool[dimension];

for (int i = 0; i < dimension; ++i) {

distance[i] = INT\_MAX;

visited[i] = false;

}

distance[edge] = 0;

for (int count = 0, index, min = INT\_MAX; count < dimension - 1; ++count, min = INT\_MAX) {

for (int i = 0; i < dimension; ++i)

if (!visited[i] && distance[i] <= min) {

min = distance[i];

index = i;

}

visited[index] = true;

for (int i = 0; i < dimension; ++i)

if (!visited[i] && graph[index][i] && distance[index] != INT\_MAX && distance[index] + graph[index][i] < distance[i])

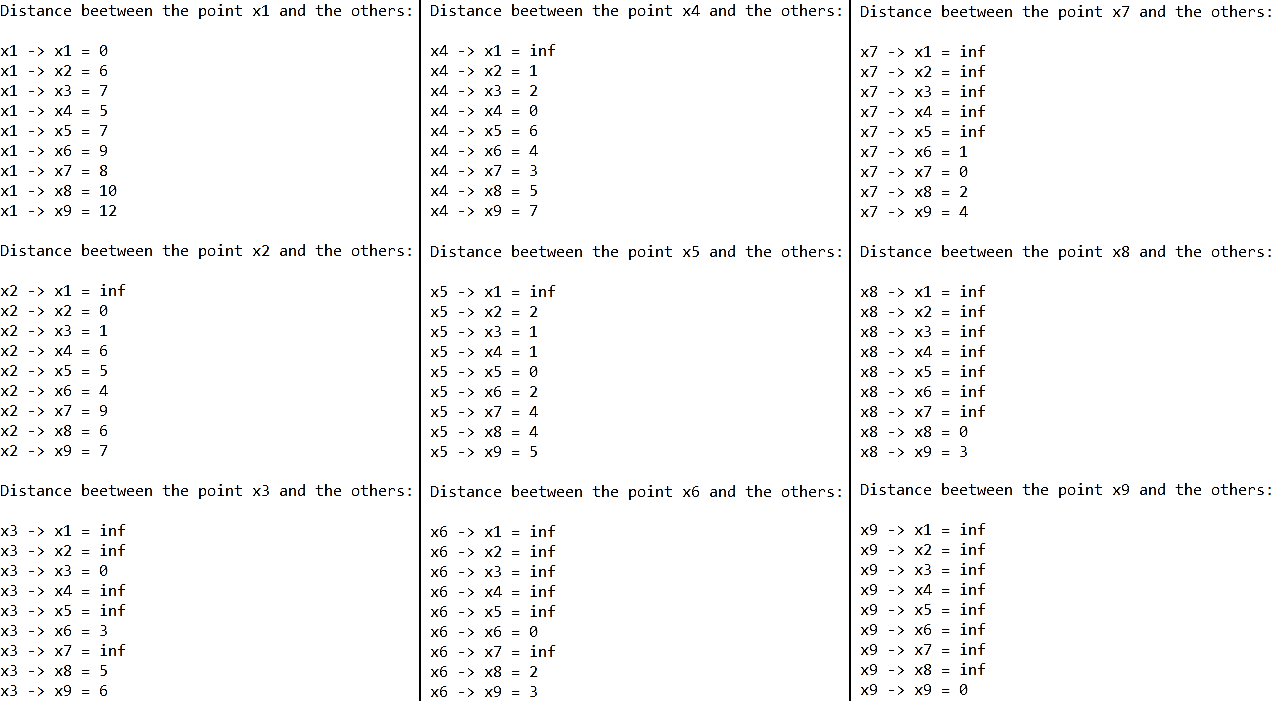
distance[i] = distance[index] + graph[index][i];

}

return distance;

}

**ВИКОНАННЯ ЕКСПЕРИМЕНТІВ ПО НАЛАГОДЖЕННЮ ПРОГРАМИ**

****

**ВИСНОВКИ**

Протягом лабораторної була розроблена программа для підрахунку ваги найкоротшого шляху між точками графа за допомогою алгоритма Дейкстри. Програма підраховує та виводить на єкран вагу найкоротшого шляху між усіма точками графа. У випадку, коли шляху немає – вага вважається нескінченною (позначка inf). Граф задається у функції main(), як матриця суміжності, у котрої єлементи зімість звичних позначок (0 та 1) використовуються 0 (у випадку, коли ребро не існує) та вага ребра (у випадку, коли ребро існує).