**МИНИСТЕРСТВО ОБРАЗОВАНИЯ РЕСПУБЛИКИ БЕЛАРУСЬ**

**УЧРЕЖДЕНИЕ ОБРАЗОВАНИЯ**

**ГОМЕЛЬСКИЙ ГОСУДАРСТВЕННЫЙ ТЕХНИЧЕСКИЙ**

**УНИВЕРСИТЕТ ИМЕНИ П. О. СУХОГО**

Факультет автоматизированных и информационных систем

Кафедра «Информационные технологии»

ЛАБОРАТОРНАЯ РАБОТА №4

по дисциплине: **«**Оптимизации проектных решений**»**

на тему:

«Алгоритм Дейкстры»

Выполнил: студент гр. ИТП-41

Столярова Ю.Ю.

Принял: преподаватель

Башаримов Ю.С.

Гомель 2023

**Цель:** знакомство с методами решения оптимизационных задач на графах.

**Задание:**

1. Изучить изложенный метод поиска кратчайшего пути (Алгоритм Дейкстры).

2. Составить программу для поиска кратчайшего пути на основе алгоритма Дейкстры.

3. Оформить отчет о выполнении задания с приведением условия задачи, алгоритма и программного кода, результаты работы программы и выводы по результатам.

**Ход работы:**

В ходе лабораторной работы была разработана программа для решения задачи о нахождении кратчайшего пути алгоритмом Дейкстры.

Выполнение программы представлено на рисунке 1.

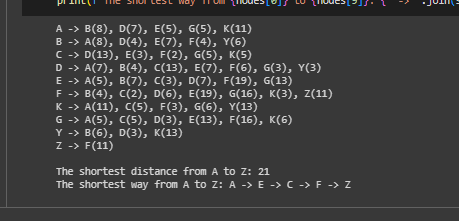
****

Рисунок 1 – Решение задачи методом Дейкстры

**Вывод:** были изучены методы решения оптимизационных задач на графах.

**ПРИЛОЖЕНИЕ А**

(обязательное)

**Текст программы**

nodes = ["A","B","C","D","E","F","K","G","Y","Z"]

init\_graph = {

nodes[0]: {nodes[1]:8, nodes[3]:7, nodes[4]:5, nodes[6]:11, nodes[7]:5},

nodes[1]: {nodes[3]:4, nodes[4]:7, nodes[5]:4, nodes[8]:6},

nodes[2]: {nodes[3]:13, nodes[4]:3, nodes[5]:2, nodes[6]:5, nodes[7]:5},

nodes[3]: {nodes[4]:7, nodes[5]:6, nodes[7]:3, nodes[8]:3},

nodes[4]: {nodes[5]:19, nodes[7]:13},

nodes[5]: {nodes[6]:3, nodes[7]:16, nodes[9]:11},

nodes[6]: {nodes[7]:6, nodes[8]:13}

}

graph = {}

for node in nodes:

graph[node] = {}

graph.update(init\_graph)

for node, edges in graph.items():

for adjacent\_node, value in edges.items():

if graph[adjacent\_node].get(node, False) == False:

graph[adjacent\_node][node] = value

print\_graph(graph)

distances, shortest\_path = dijkstra(graph, nodes[0], nodes[9])

print(f"\nThe shortest distance from {nodes[0]} to {nodes[9]}: {distances[nodes[9]]}")

print(f"The shortest way from {nodes[0]} to {nodes[9]}: {' -> '.join(shortest\_path)}")