Министерство образования и науки РФ

Федеральное государственное автономное образовательное учреждение высшего образования

«Омский государственный технический университет»

|  |  |
| --- | --- |
| Факультет (институт) | *Информационных технологий и компьютерных систем* |
|  |  |
| Кафедра | *Прикладная математика и фундаментальная информатика* |
|  |  |

**Расчетно-графическая работа**

|  |  |
| --- | --- |
| по дисциплине | ***Дискретная математика*** |
|  |  |
| на тему | Разработка программы |

Пояснительная записка

|  |  |
| --- | --- |
| **Шифр проекта** | 020-РГР-02.03.02-№ 9-ПЗ |

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  |  |  | **Студента** | | Есакова Кирилла Николаевича | | | | | |
|  |  |  |  | | фамилия, имя, отчество полностью | | | | | |
|  |  |  | Курс | *1* |  | Группа | | ФИТ-232 | | |
|  |  |  |  |  |  | |  |  | |  |
|  | | | **Направление (специальность)** | | | | | ***02.03.02*** | | |
|  | | | *Фундаментальная информатика и информационные технологии* | | | | | | | |
|  |  |  | код, наименование | | | | | | | |
|  |  |  | Руководитель | | ***ст. преподаватель*** | | | | | |
|  |  |  | ученая степень, звание | | | | | |
|  |  |  | ***Федотова И.В.*** | | | | | | | |
|  |  |  | фамилия, инициалы | | | | | | | |
|  |  |  | Выполнил | | 01.06.2024 | | | | | |
|  |  |  | дата, подпись студента | | | | | |
|  |  |  | **Работа защищена с количеством баллов** | | | | | | | |
|  |  |  |  | | | | | |  | |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  |  | дата, подпись руководителя |  |  |  |

Омск 2024

Содержание

[Введение 3](#_Toc167620494)

[Теоретическая часть 4](#_Toc167620495)

[Разработка кода 6](#_Toc167620496)

[Результаты программы 8](#_Toc167620497)

[Заключение 10](#_Toc167620498)

[Список используемой литературы 11](#_Toc167620499)

# **Введение**

В ходе работы необходимо написать программу, которая с помощью алгоритма Флойда Уоршелла найдет минимальное расстояние между всеми парами вершин. Описание задачи:

На графе изображена система связи предприятий (см. Рисунок1). Вероятность утечки информации при передаче (в процентах) соответствует весу рij ребра (xi, xj). Найти минимальную величину утечки информации при ее передаче между всеми предприятиями.

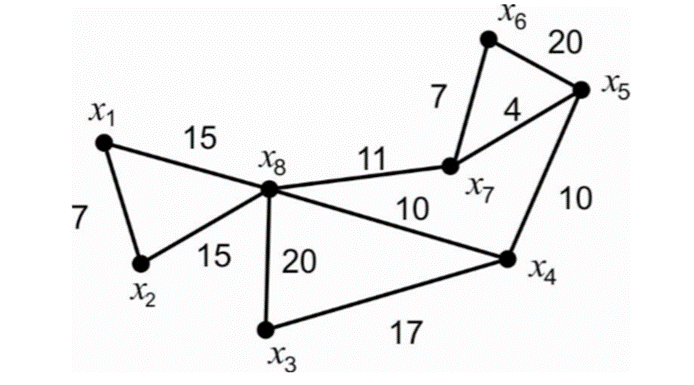


Рисунок 1 – Исходный граф

**Формат входных данных**

Во входном файле записано сначала число N (1 <= N <=100), определявшее количество точек передачи информации. Затем идет число M – количество соединений между точками передачи, далее идет описание потерь информации для каждой линии передачи. Каждое соединение между точками задается тремя числами – номерами точек передачи информации и величиной утечки информации.

**Формат выходных данных**

На экран выведите числа – минимальные значение утечки информации с указанием номеров пунктов или -1, если передача информации отсутствует.

# **Теоретическая часть**

Алгоритм Флойда Уоршелла – динамический алгоритм вычислений значений кратчайших путей для каждой пары вершин графа. Метод работает на взвешенных графах, с положительными и отрицательными весами ребер, но без отрицательных циклов и является, таким образом, более общим, чем алгоритм Дейкстры, который не работает с отрицательными весами и, в классической своей версии, подразумевает определение оптимальных расстояний только от одной вершины до всех остальных

Для реализации алгоритма Флойда необходимо сформировать матрицу смежности исходного графа G (V, E), но каждому элементу матрицы присваивается вес ребра из вершины i в вершину j.

По мере выполнения алгоритма данная матрица будет перезаписываться. И в каждую её ячейку будут заноситься значения путей оптимальной длины из вершины в вершину.

Пусть дана матрица весов дуг ориентированного графа G (V, E), необходимо определить кратчайшие расстояния между всеми парами вершин (vi,vj).

**Описание алгоритма**

1. Присвоение начальных значений.

Устанавливается номер итерации k = 0. Строится матрица смежности.

1. Итерационный процесс.

Присваивается k=k+1. Для всех i, удовлетворяющих условию, что и для всех j, таких, что, , вычислить новые значения

Все другие элементы матрицы Dсохранить без изменений, т. е.

3. Проверка условия окончания

а) Если, то граф G содержит контур отрицательной длины, проходящий через вершину vi, и задача не имеет решения. Конец алгоритма.

б) Если и k=n, где n – количество вершин в графе, то решение получено. Конец алгоритма.

в) Если и k <n, то вернуться к п. 2.

Рассмотрим работу алгоритма на примере нашей задачи.

При вычислении кратчайших путей, проходящих через пункт x1, матрица не изменится, следовательно

При вычислении кратчайших путей, проходящих через пункты x1, x2, матрица не изменится, следовательно

При вычислении кратчайших путей, проходящих через пункты x1, x2, x3, матрица не изменится, следовательно

На четвертой итерации появляются кратчайшие пути

На пятой итерации появляются кратчайшие пути

При вычислении кратчайших путей, проходящих через пункты x1, x2, x3, x4 x5 x6 матрица не изменится, следовательно

На седьмой итерации появляются кратчайшие пути

На восьмой итерации появляются кратчайшие пути

В итоге получаем матрицу кратчайших расстояний между всеми предприятиями.

# **Разработка кода**

def perform\_calculation(graph):  
 n = len(graph)  
 weights = [[float('inf')] \* n for \_ in range(n)]  
  
 for i in range(n):  
 weights[i][i] = 0  
  
 for start, neighbors in graph.items():  
 for end, weight in neighbors.items():  
 weights[int(start) - 1][int(end) - 1] = weight  
  
 for k in range(n):  
 for i in range(n):  
 for j in range(n):  
 weights[i][j] = min(weights[i][j], weights[i][k] + weights[k][j])  
  
 print("Кратчайшие расстояния между всеми парами вершин:")  
 for i in range(n):  
 for j in range(i+1, n):   
 print(f"Из пункта №{i+1} - В пункт №{j+1}: минимальное значение утечки информации {weights[i][j]}")  
  
  
def show\_student\_info():  
 print("Есаков Кирилл Николаевич, ФИТ-232/2, 1 курс")  
 input("Нажмите Enter, чтобы вернуться в меню")  
  
def show\_task\_description():  
 with open('RgrTask.txt', 'r',encoding="utf-8") as file:  
 task = file.read()  
 print(task)  
 input("Нажмите Enter, чтобы вернуться в меню")  
def main():  
 filename = "input.txt"  
 with open(filename, "r") as f:  
 num\_vertices = int(f.readline())  
 num\_edges = int(f.readline())  
  
 graph = {str(i): {} for i in range(1, num\_vertices + 1)}  
  
 for \_ in range(num\_edges):  
 start, end, weight = map(int, f.readline().split())  
 graph[str(start)][str(end)] = weight  
 graph[str(end)][str(start)] = weight  
  
 while True:  
 print("Меню:")  
 print("1) Информация о студенте")  
 print("2) Описание задачи")  
 print("3) Вычисление")  
 print("4) Выход из программы")  
  
 choice = input("Выберите пункт меню: ")  
  
 if choice == '1':  
 print("-"\*100)  
 show\_student\_info()  
 print("-" \* 100)  
  
 elif choice == '2':  
 print("-" \* 100)  
 show\_task\_description()  
 print("-" \* 100)  
  
 elif choice == '3':  
 print("-" \* 100)  
 perform\_calculation(graph)  
 print("-" \* 100)  
  
 elif choice == '4':  
 print("Выход из программы")  
 break  
  
 else:  
 print("Некорректный выбор. Попробуйте снова.")  
  
if \_\_name\_\_ == "\_\_main\_\_":  
 main()

# **Результаты программы**

Рассмотрим результаты программы:

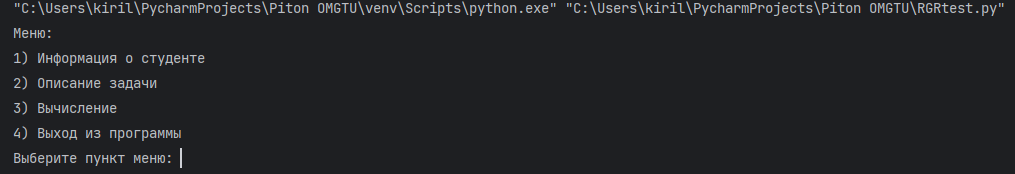


Рисунок 2 – Меню программы

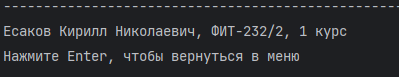


Рисунок 3 – Пункт «Информация о студенте»

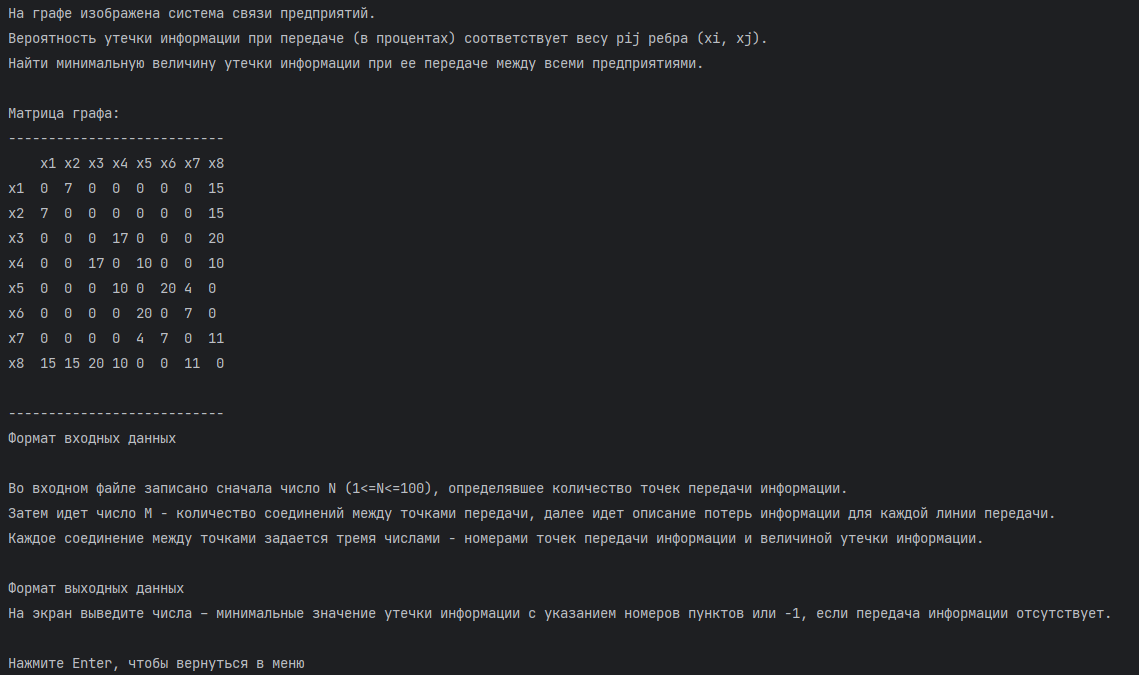


Рисунок 4 – Пункт «Описание задачи»

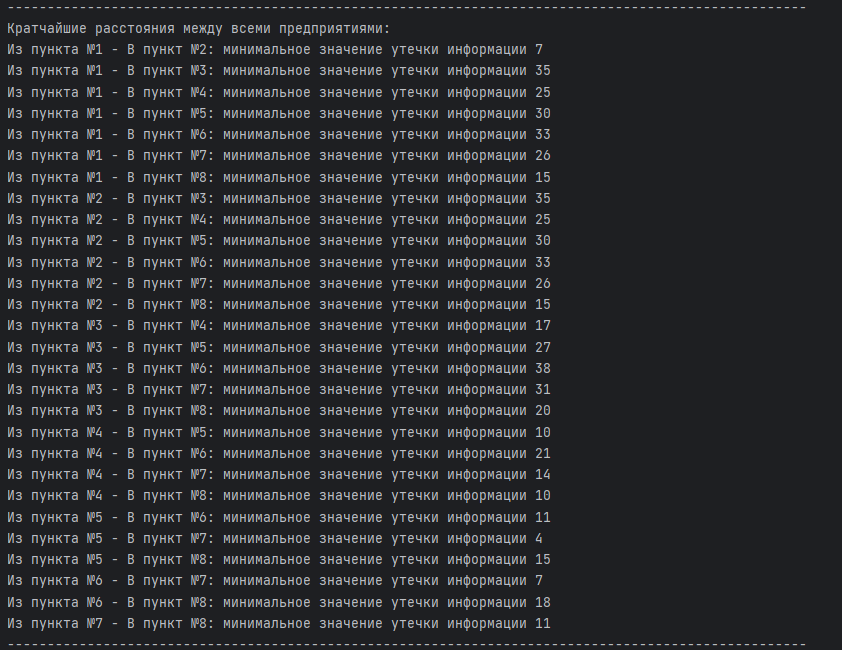


Рисунок 5 – Пункт «Вычисление»

# **Заключение**

В результате работы мы разработали программу, которая с помощью алгоритма Флойда-Уоршелла находит кратчайшее расстояние между всеми предприятиями. Данная работа помогла закрепить знания, полученные во время изучения дисциплины «Дискретная математика».

# **Список используемой литературы**

1. Metanit, Руководство по языку программирования Python, <https://metanit.com/python/tutorial/> – (дата обращение: 01.06.2024)
2. *Применение алгоритмов на графах для решения прикладных задач*: метод. указания /Минобрнауки России, Омский государственный технический университет/; Е. А. Калиберда, И. В. Федотова – Омск: Изд-во ОмГТУ, 2021. (дата обращения: 01.06.2024)