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 по поиску минимального пути |

Пояснительная записка

|  |  |
| --- | --- |
| **Шифр проекта** | 020-РГР-02.03.02-№ 25-ПЗ |

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  |  |  | **Студента** | | Харлов Никита Станиславович | | | | | |
|  |  |  |  | | 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 | |  | | | | | |
|  |  |  | дата, подпись студента | | | | | |
|  |  |  | **Работа защищена с количеством баллов** | | | | | | | |
|  |  |  |  | | | | | |  | |

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

Омск 2024

**Оглавление**

[Введение 3](#_Toc168433990)

[Задача 4](#_Toc168433991)

[Алгоритм Форда-Беллмана 5](#_Toc168433992)

[Код программы 6](#_Toc168433993)

[Результаты 8](#_Toc168433994)

[Заключение 10](#_Toc168433995)

[Список используемой литературы 11](#_Toc168433996)

# **Введение**

В вихре неустанных перемен, информационные технологии (ИТ) превратились в неотъемлемую силу, пронизывающую каждый аспект человеческого бытия. От рождения интернета до революции мобильных устройств.

ИТ эволюционировали от полезных инструментов до движущей силы, тесно переплетенной с нашей повседневной жизнью. ИТ оказывают глубокое влияние на ключевые сферы общества: экономику, образование, здравоохранение, коммуникации, науку и многое другое. Они оптимизируют операции, расширяют доступ к информации, улучшают сотрудничество и способствуют инновациям.

В частности, благодаря ИТ, реализация математических алгоритмов стала гораздо проще, поскольку теперь огромные расчеты можно делегировать компьютерам, предоставив им соответствующие программы.

# **Задача**

**Задача 25**

Дан ориентированный взвешенный полный граф, рёбрам которого приписаны некоторые веса (длины). Веса могут быть и положительные, и отрицательные, и нулевые. Нас интересует минимум длин всех возможных путей между всеми парами различных вершин этого графа. Нужно будет выяснить, существует ли этот минимум, и, если существует, вычислить его. (Минимума не существует в том случае, если в графе можно найти путь отрицательной длины, сколь угодно большой по модулю).

**Формат входных данных**

В первой строке входного файла задано число вершин N (3≤ N ≤ 50). Далее идёт матрица смежности графа, то есть N строк, в каждой из которых записано N чисел. ј-ое число в 1-ой строке матрицы смежности задает длину ребра, ведущего из і-й вершину в ј-ую. Длины могут принимать значения, не превосходящие 106 по абсолютной величине. Гарантируется, что на главной диагонали матрицы стоят нули.

**Формат выходных данных**

В выходной файл вывести искомый минимум. Если его не существует, вывести -1.

# **Алгоритм Форда-Беллмана**

Алгоритм Форда-Беллмана — это метод поиска кратчайших путей в взвешенном графе, который может содержать рёбра с отрицательным весом. Однако, если в графе есть циклы отрицательного веса, достижимые из источника, алгоритм сообщит о их наличии. Вот основные шаги алгоритма:

1. **Инициализация**: Установите расстояние до начальной вершины равным 0, а расстояния до всех остальных вершин равными бесконечности.
2. **Релаксация рёбер**: Проходите по всем рёбрам графа и для каждого ребра (𝑢,𝑣) с весом 𝑤, если расстояние до вершины 𝑢плюс вес ребра меньше расстояния до вершины 𝑣, обновите расстояние до 𝑣: 𝑑𝑖𝑠𝑡[𝑣]=𝑑𝑖𝑠𝑡[𝑢]+𝑤.
3. **Повторение**: Шаг релаксации выполняется ∣𝑉∣−1 раз, где ∣𝑉∣ — количество вершин в графе. Это гарантирует, что кратчайшие пути будут найдены, даже если они проходят через все вершины.
4. **Проверка на циклы отрицательного веса**: Проходите по всем рёбрам ещё раз и проверьте, не уменьшится ли расстояние до какой-либо вершины после ∣𝑉∣−1 итераций. Если расстояние уменьшается, в графе есть цикл отрицательного веса.

Алгоритм Форда-Беллмана полезен в ситуациях, когда необходимо учитывать рёбра с отрицательным весом, и он часто используется в задачах оптимизации и сетевого планирования. Временная сложность алгоритма составляет 𝑂(∣𝑉∣⋅∣𝐸∣), где ∣𝐸∣ — количество рёбер в графе. Это делает его менее эффективным для плотных графов по сравнению с алгоритмом Дейкстры, но он способен обрабатывать графы, в которых алгоритм Дейкстры не применим из-за отрицательных весов рёбер.

# **Код программы**

Код программы на языке Python:

def bellman\_ford(graph, source):

vertices = set()

for edge in graph:

vertices.add(edge[0])

vertices.add(edge[1])

distances = {vertex: float('infinity') for vertex in vertices}

distances[source] = 0

for \_ in range(len(vertices) - 1):

for edge in graph:

u, v, weight = edge

if distances[u] + weight < distances[v]:

distances[v] = distances[u] + weight

for edge in graph:

u, v, weight = edge

if distances[u] + weight < distances[v]:

return -1

return distances

def execute\_Bellman():

sums = 0

with open("./Calculating.txt") as file:

n = int(file.readline())

graph = []

for i in range(n):

l = list(map(int, file.readline().split(" ")))

for j in range(n):

if l[j] != 0:

graph.append([i, j, l[j]])

for i in range(n):

ep = bellman\_ford(graph, i).values()

for i in ep:

sums += i

print(sums)

f = open("output.txt", "w")

f.write(str(sums))

f.close()

return

def Menu():

while True:

print("--Меню--")

print("1. Запуск алгоритма Форда-Беллмана")

print("2. Об авторе")

print("3. Описание задачи")

print("4. Выход")

choose = int(input())

match choose:

case 1:

execute\_Bellman()

case 2:

print("Выполнил\nСтудент группы Фит-232 \nХарлов Никита Станиславович ")

case 3:

print("""

Дан ориентированный взвешенный полный граф, рёбрам которого приписаны

некоторые веса (длины). Веса могут быть и положительные, и отрицательные, и

нулевые. Нас интересует минимум длин всех возможных путей между всеми парами

различных вершин этого графа. Нужно будет выяснить, существует ли этот минимум,

и, если существует, вычислить его. (Минимума не существует в том случае, если в

графе можно найти путь отрицательной длины, сколь угодно большой по модулю).

Формат входных данных

В первой строке входного файла задано число вершин N (3≤ N ≤ 50). Далее идёт

матрица смежности графа, то есть N строк, в каждой из которых записано N чисел. ј-ое

число в 1-ой строке матрицы смежности задает длину ребра, ведущего из і-й вершину в

ј-ую. Длины могут принимать значения, не превосходящие 106 по абсолютной

величине. Гарантируется, что на главной диагонали матрицы стоят нули.

Формат выходных данных

В выходной файл вывести одно число искомый минимум. Если его не

существует, вывести -1"""

)

case 4:

break

if \_\_name\_\_ == '\_\_main\_\_':

Menu()

# **Результаты**

На рисунке 1 показано меню программы.

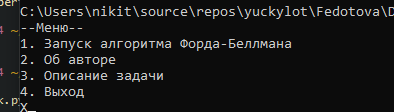


Рисунок 1 – Меню программы

При выборе «Запуск алгоритма Форда-Беллмана» в консоль выводится минимальный путь и он же записывается в файл, что показано на рисунке 2.

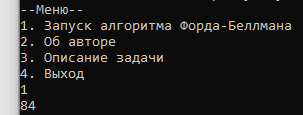


Рисунок 2 – Результат выбора первого пункта

При выборе «Об авторе» в консоль выводится информация об авторе, что показано на рисунке 3.

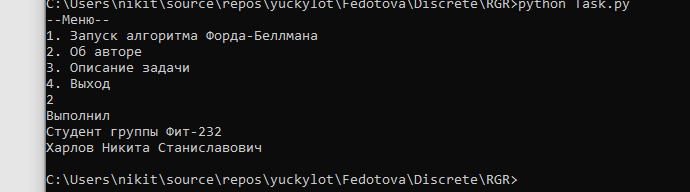


Рисунок 3 – Результат выбора второго пункта

При выборе «Описание задачи» в консоль выводится информация о задаче, что показано на рисунке 4.

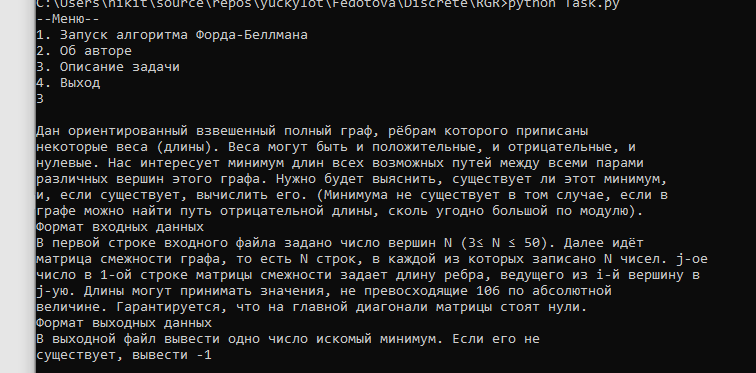


Рисунок 4 – Результат выбора второго пункта

При выборе «Выход» программа завершает свою работу.

# **Заключение**

В ходе выполнения расчетно-графической работы был изучен алгоритм нахождения минимального пути (алгоритм Форда-Беллмана) и на основе полученных знаний была решена задача.

# **Список используемой литературы**

1. Минимальный путь \\ Википедия – URL: <https://en.wikipedia.org/wiki/Shortest_path_problem> (дата обращения 03.06.2024)

2. Алгоритм Форда-Беллмана \\ Википедия – URL: <https://en.wikipedia.org/wiki/Bellman%E2%80%93Ford_algorithm> (дата обращения 03.06.2024)