

МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ

Федеральное государственное бюджетное образовательное учреждение высшего образования

**"МИРЭА - Российский технологический университет"**

**РТУ МИРЭА**

|  |
| --- |
| **Институт информационных технологий (ИТ)** |
| **Кафедра математического обеспечения и стандартизации информационных технологий (МОСИТ)** |

**ОТЧЁТ ПО ПРАКТИЧЕСКОМУ ЗАДАНИЮ №5**

**по дисциплине «Структуры и алгоритмы обработки данных»**

Тема: **«**Основные алгоритмы работы с графами.»

|  |  |  |  |
| --- | --- | --- | --- |
| Отчет представлен к рассмотрению:  Студент группы ИНБО-01-20 | «1» ноября 2021 г. |  | Салов В.Д. |
| (подпись) | | | |
| Преподаватель | «1» ноября 2021 г. |  | Сорокин А.В. |
|  | (подпись) | | |

Москва, 2021 г.

CОДЕРЖАНИЕ

[Цель работы 3](#_Toc86720308)

[Постановка задачи 3](#_Toc86720309)

[Подход к решению. 3](#_Toc86720310)

[Алгоритмы операций на псевдокоде. 5](#_Toc86720311)

[Код программы. 7](#_Toc86720312)

[Тестирование программы. 10](#_Toc86720313)

[Вывод 14](#_Toc86720314)

[Список информационных источников 15](#_Toc86720315)

# Цель работы

Получение практических навыков по выполнению операций над структурой данных «граф».

# Постановка задачи

Выполнить разработку программы управления графом, в соответствии с вариантом, на основе класса «Граф». Предусмотреть в качестве данных: количество вершин в графе, структура для хранения графа.

**Вариант 9.**

|  |  |
| --- | --- |
| Представление графа в памяти | Задачи варианта |
| Матрица смежности | Ввод с клавиатуры графа (применение операции вставки ребра в граф).  Определить медиану неориентированного графа.  Составить программу нахождения кратчайшего пути в графе от заданной вершины к другой заданной вершине методом «Дейкстры» и вывести этот путь. |

**Дано:**

Произвольный граф (ориентированный или неориентированный, связный или несвязный) с известным количеством вершин.

**Результат.**

Отображение графа в виде матрицы смежности.

Реализованные операции варианта.

## Подход к решению.

1. В ходе работы был разработан класс графа, реализующий, согласно варианту, следующие методы: создание графа посредством применения операций вставки ребра в граф, вывод матрицы смежности графа, определение медианы графа, нахождение величины кратчайшего пути и вывод кратчайшего пути от заданной вершины к другой заданной вершине методом «Дейкстры».
2. Разработан класс узла графа, содержащий информационную часть и поле индекса.
3. Разработан консольный пользовательский интерфейс для тестирования работоспособности программы.
4. Разработаны методы обработки графа:
   1. Метод вставки ребра в граф – добавление ребра с заданным весом между двумя заданными вершинами;
   2. Метод получения индекса узла – возврат индекса для заданного узла графа;
   3. Метод определения медианы графа – возврат индекса вершины, для которой сумма кратчайших путей до других вершин минимальна.
   4. Метод «Дейкстры» – поиск величин кратчайших путей до каждой вершины для заданной вершины;
   5. Метод восстановления кратчайшего пути – возврат кратчайшего маршрута из одной заданной вершины в другую;
   6. Метод нахождения величины кратчайшего пути – возврат величины кратчайшего пути между двумя заданными вершинами графа, найденного с помощью метода алгоритма Дейкстры.
5. Разработаны методы приложения для тестирования:
   1. Метод для тестирования вывода графа – организация ввода графа и вывода его матрицы смежности;
   2. Метод для тестирования нахождения кратчайшего пути – организация нахождения кратчайшего пути от одной заданной вершины к другой с помощью алгоритма Дейкстры и вывода результатов работы алгоритмов в консоль;
   3. Метод для тестирования определения медианы графа – организация определения медианы графа и вывода результата работы алгоритма в консоль.

## Алгоритмы операций на псевдокоде.

**Метод вставки ребра в граф:**

процедура connect(первая\_вершина, вторая\_вершина, вес := 1):  
 первая\_вершина := self.get\_index(первая\_вершина)

вторая\_вершина := self.get\_index(вторая\_вершина)  
 матрица\_смежности[первая\_вершина][вторая\_вершина] := вес

**Метод получения индекса узла:**

функция get\_index(узел):  
 если принадлежность(узел, int):  
 возврат узел  
 иначе:  
 возврат узел.индекс

**Метод «Дейкстры»:**

функция dijkstra(узел):  
 узел := get\_index(узел)  
 из collections импортировать default\_словарь  
 граф := default\_словарь(список)  
 для каждого row от 0 до длина(матрица\_смежности) - 1:  
 для каждого col от 0 до длина(матрица\_смежности) - 1:  
 если матрица\_смежности[row][col] есть не None и матрица\_смежности[row][col] != 0:  
 граф[row] := граф[row] + [(матрица\_смежности[row][col], col)]  
 nodes\_to\_visit := массив[]  
 nodes\_to\_visit.добавить\_элемент\_в\_конец((0, узел))  
 visited := множество()  
 min\_dist := {i: ∞ для каждого i от 0 до длина(матрица\_смежности) - 1}  
 min\_dist[узел] := 0  
 пока длина(nodes\_to\_visit) > 0:

вес, текущий\_узел := минимальный\_элемент(nodes\_to\_visit)  
 nodes\_to\_visit.удалить\_элемент((вес, текущий\_узел))  
 если текущий узел в visited:  
 принудительный\_запуск\_следующего\_прохода\_цикла  
 visited.добавить\_элемент(текущий\_узел)  
 для след\_вес, след\_узел в graph[текущий\_узел]:  
 если вес + след\_вес < min\_dist[след\_узел] и след\_узел не в visited:  
 min\_dist[след\_узел] := вес + след\_вес  
 nodes\_to\_visit.добавить\_элемент\_в\_конец((вес + след\_вес, след\_узел))  
 возврат min\_dist

**Метод восстановления кратчайшего пути:**

функция path\_restoring(узел1, узел2):  
 visited := [None] \* длина(матрица\_смежности)  
 узел1 := self.get\_index(узел1)  
 узел2 := self.get\_index(узел2)  
 visited[0] := узел2 + 1  
 пред\_индекс := 1  
 вес := shortest\_path(узел1, узел2)  
 пока узел2 != узел1:  
 для каждого i от 0 до длина(матрица\_смежности) - 1:  
 если матрица\_смежности[i][узел2] есть не None и матрица\_смежности[i][узел2] != 0:  
 temp := вес - матрица\_смежности[i][узел2]  
 если temp == shortest\_path(узел1, i):  
 вес := temp  
 узел2 := i  
 visited[пред\_индекс] := i + 1  
 пред\_индекс := пред\_индекс + 1

пока None в visited:

visited.удалить\_элемент(None)  
 возврат развернуть\_массив(visited)

**Метод нахождения величины кратчайшего пути:**

функция shortest\_path(узел1, узел2):  
 узел2 := get\_index(узел2)  
 возврат dijkstra(узел1)[узел2]

**Метод определения медианы графа:**

функция median():  
 array := массив[]  
 для каждого узел1 от 0 до длина(матрица\_смежности) - 1:  
 сумма\_путей := 0  
 для каждого узел2 от 0 до длина(матрица\_смежности) - 1:  
 сумма\_путей := сумма\_путей + shortest\_path(узел1, узел2)  
 array.добавить\_элемент\_в\_конец(сумма\_путей)  
 возврат array.индекс(минимальный\_элемент(array))

## Код программы.

Класс узла графа:

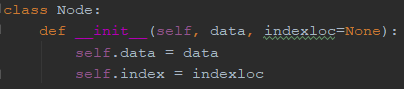


Рисунок 1 – Класс узла графа.

Класс графа:



Рисунок 2 – Класс графа.



Рисунок 3 – Класс графа (продолжение).

Основная функция для тестирования:

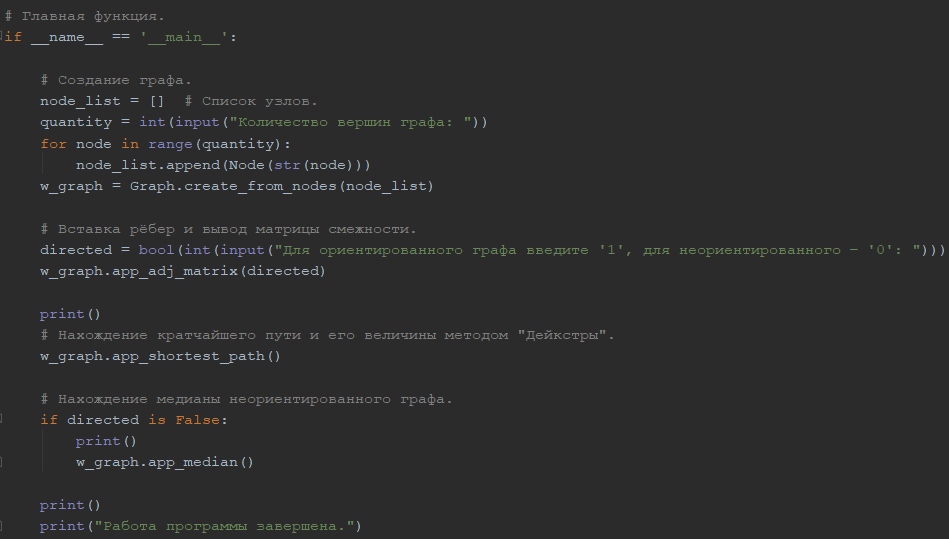


Рисунок 4 – Основная функция.

## Тестирование программы.

Рассмотрим в качестве примера следующий ориентированный граф:

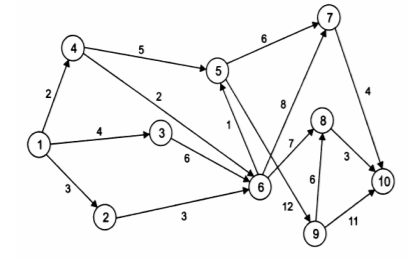


Рисунок 5 – Ориентированный граф для тестирования программы.

Было запущено тестирование: через консоль введено количество вершин графа, тип графа; были произведены операции вставки ребра в граф.

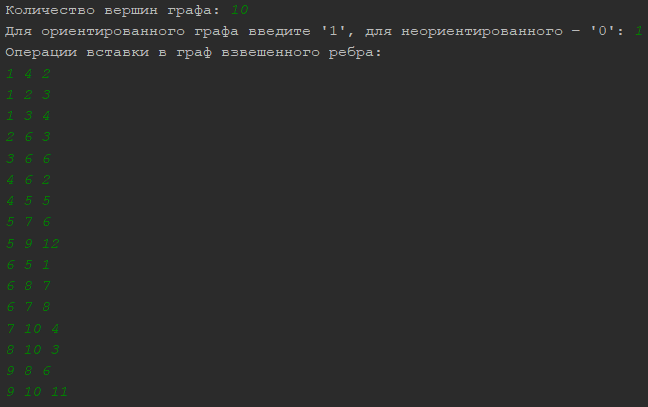


Рисунок 6 – Ввод графа посредством операций вставки ребра в граф.

В результате был выведен граф в виде матрицы смежности, в которой каждая строка является набором длин исходящих путей в каждую вершину графа для соответствующей вершины графа. Путь от любой вершины до самой себя равен 0. Если с какой-либо вершиной отсутствует прямая связь, то в ячейке хранится None.

Результат работы программы:

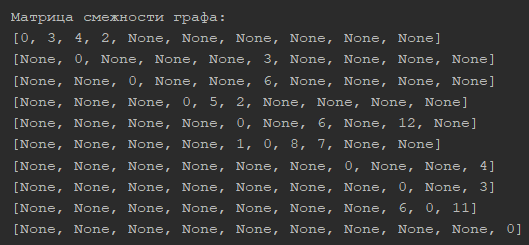


Рисунок 7 – Вывод матрицы смежности графа.

Далее вводятся две вершины графа для нахождения величины кратчайшего пути методом «Дейкстры» и вывода кратчайшего пути от первой заданной вершины ко второй.

Для тестирования были взяты вершины «1» и «10». По рисунку видно, что кратчайшим путём от вершины «1» к вершине «10» является путь: 1 – 4 – 6 – 8 – 10. Величина этого пути: 14.

Результат работы программы подтверждает полученный результат:

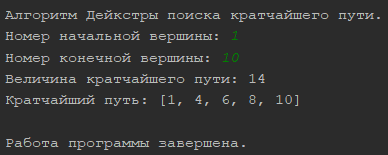


Рисунок 8 – Вывод кратчайшего пути и его величины методом «Дейкстры».

Теперь пусть дан неориентированный граф следующего вида:

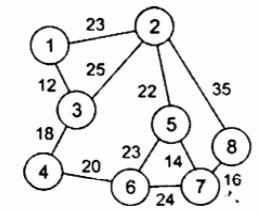


Рисунок 9 – Неориентированный граф для тестирования программы.

Было запущено тестирование: через консоль введено количество вершин графа, указан тип графа, произведены операции вставки ребра в граф.

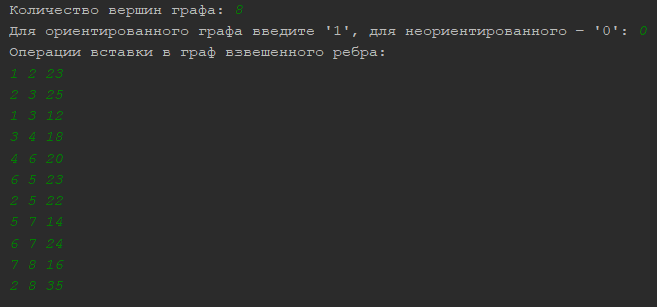


Рисунок 10 – Ввод графа посредством операций вставки ребра в граф.

В результате была выведена матрица смежности для данного графа.

Результат работы программы:

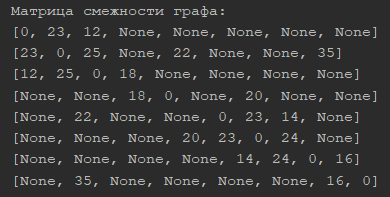


Рисунок 11 – Вывод матрицы смежности графа.

Найдём кратчайший путь из вершины «5» к вершине «1» и его величину методом «Дейкстры».

По графу видно, что кратчайшим путём от вершины «5» к вершине «1» является путь: 5 – 2 – 1. Его величина: 45.

Результат работы программы это подтверждает:

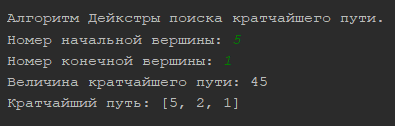


Рисунок 12 – Вывод кратчайшего пути и его величины методом «Дейкстры».

Так как граф неориентированный, найдём для него медиану:

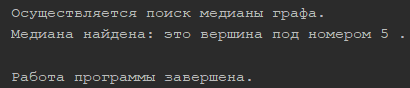


Рисунок 13 – Определение медианы графа.

Тестирование пройдено успешно: все задачи варианта выполнены; программа работает правильно.

# Вывод

В ходе работы были приобретены умения и навыки разработки и реализации операций над структурой данных «граф».

# Список информационных источников

1. Лекции по дисциплине «Структуры и алгоритмы обработки данных» /

Л. А. Скворцова, МИРЭА – Российский технологический университет, 2021.