Министерство образования и науки Российской Федерации

Федеральное государственное бюджетное образовательное учреждение высшего образования

«**Пермский национальный исследовательский политехнический университет»**

Кафедра «Информационные технологии и автоматизированные системы»

**ОТЧЕТ**

"Графы"

Выполнил работу

Студент группы РИС-23-3Б

Епин Т. Е.

Проверил

Доцент кафедры ИТАС

Полякова О.А.

Г. Пермь-2023

# Постановка задачи

Реализовать следующие алгоритмы для собственного варианта графа, имеющего не менее 6 вершин:

1. Обход в ширину;
2. Обход в глубину;
3. Алгоритм Дейкстры.

Требования:

1. Пользовательский интерфейс на усмотрение разработчика с условием кроссплатформенности;
2. Визуализация графа с использованием любой доступной графической библиотеки;
3. Реализованные алгоритмы должны справляться как с графом, представленным в задании варианта, так и с другими на усмотрение проверяющего;
4. Необходимо реализовать функции для редактирования графа:

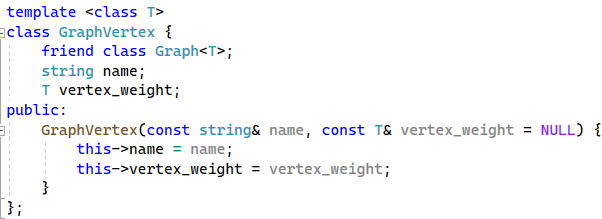
* Создание новой вершины;
* Удаление вершины;
* Добавление и удаление ребра;
* Редактирование весов ребер;

# Описание классов GraphVertex и Graph

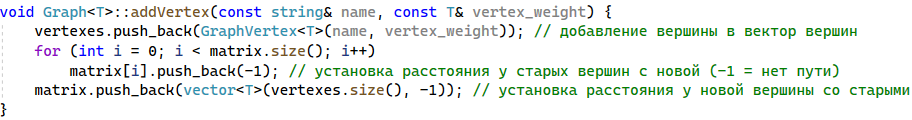
Для решения всех поставленных задач необходимо реализовать отдельный параметризованный класс Graph. Данный класс будет содержать матрицу смежности, реализованную как вектор векторов и вектор вершин графа:

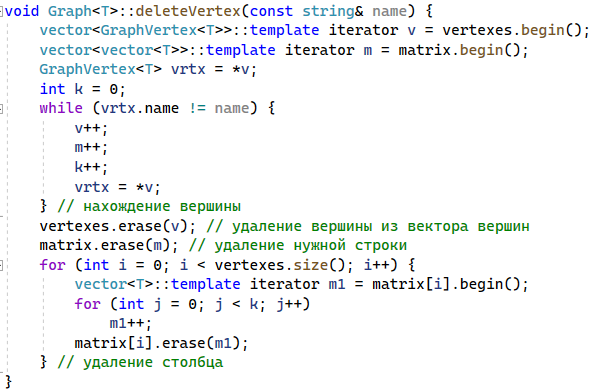


Для вектора вершин необходимо реализовать отдельный класс вершины графа:

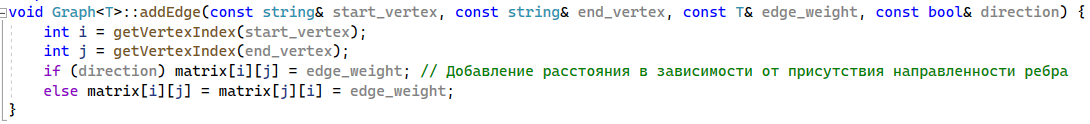


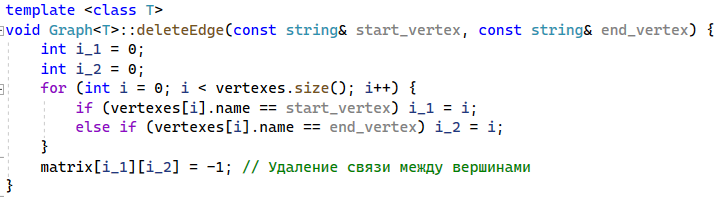
Добавление и удаление вершин происходит следующим образом:



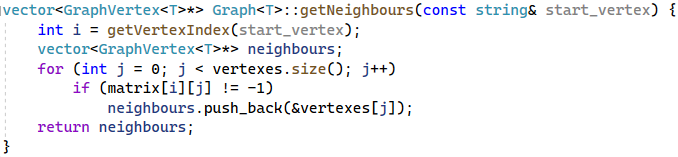


Добавление и удаление рёбер осуществляется с помощью работы с матрицей смежности:

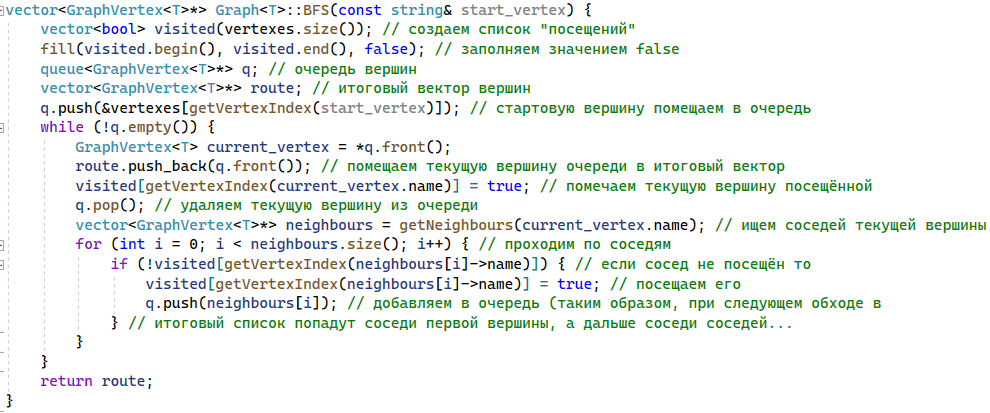




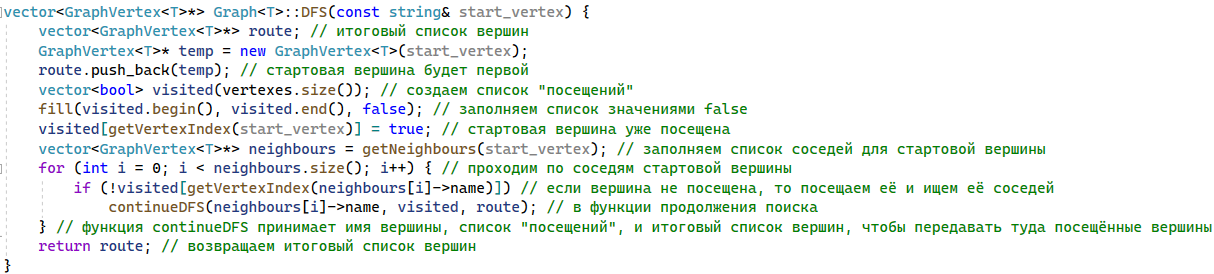
Для алгоритмов необходима функция по поиску соседей текущей вершины getNeighbours, возвращающая вектор вершин:

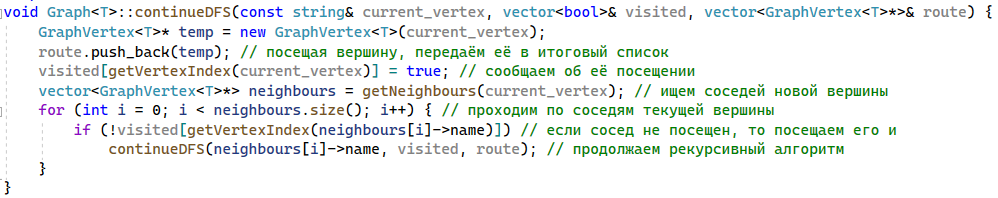


Алгоритм обхода в ширину принимает имя стартовой вершины и возвращает вектор вершин. Последовательность этих вершин и является ответом для поиска в ширину:

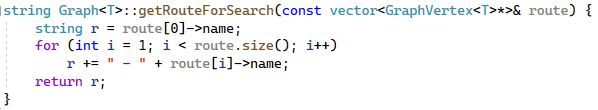


Алгоритм обхода в глубину также принимает стартовую вершину и возвращает вектор вершин:

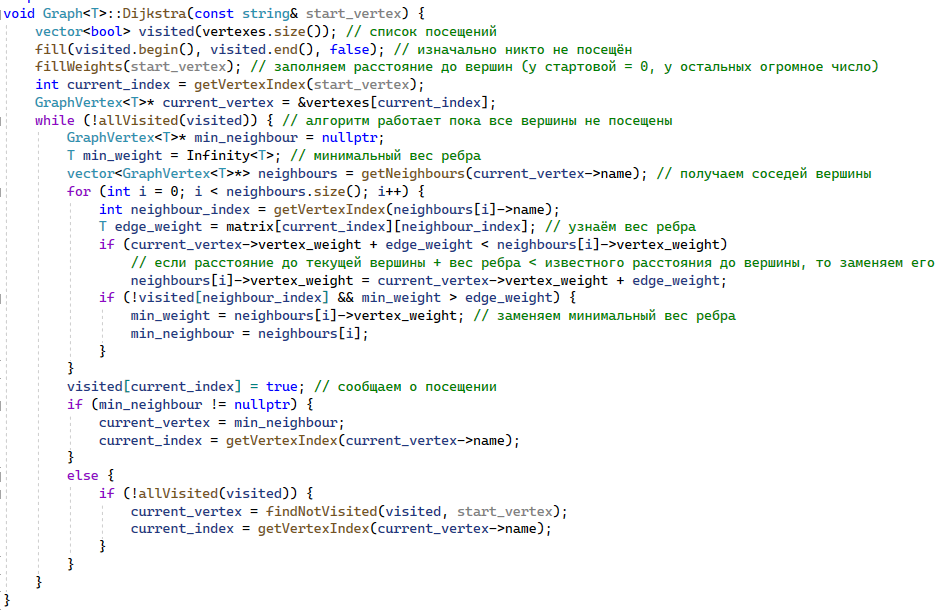




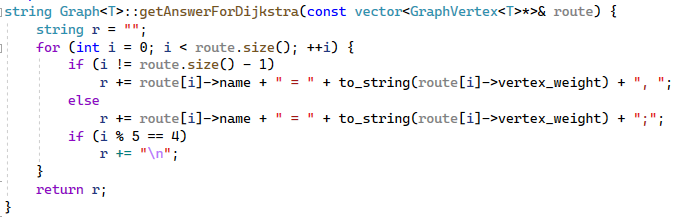
Для вывода ответа поисковых алгоритмов необходима следующая функция:



Алгоритм Дейкстры заполняем расстояние до вершин внутри самих вершин:



Для вывода ответа необходима следующая функция:

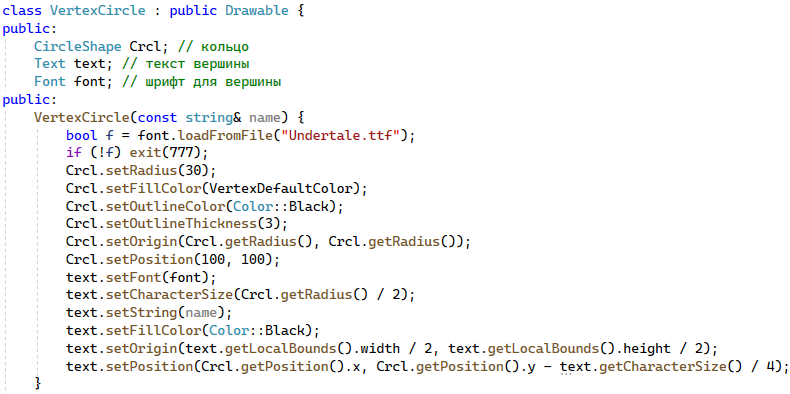


# Визуализация графа

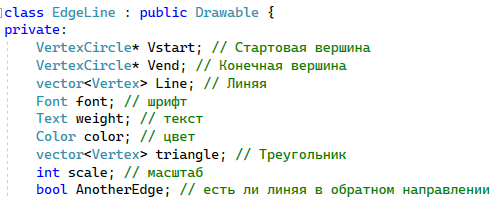
Визуализация графа выполнена с помощью графической библиотеки SFML. Для реализации были написаны следующие классы:

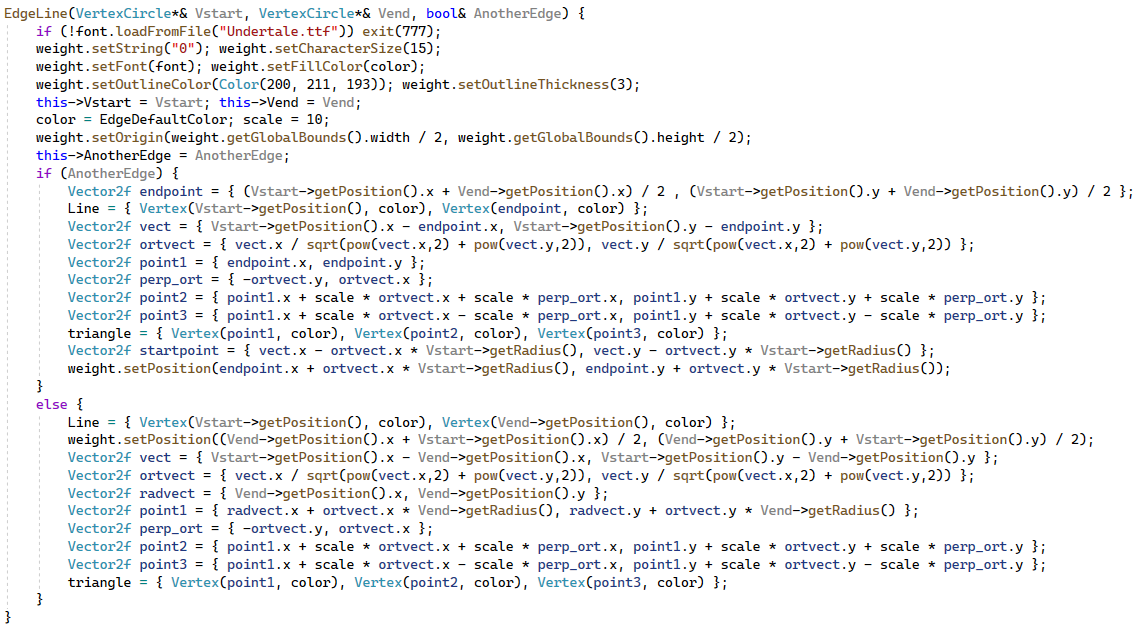
1. VertexCircle - вершина графа, реализованная с помощью кольца. Сам класс наследуем от класса Drawable, определенного в библиотеке SFML.

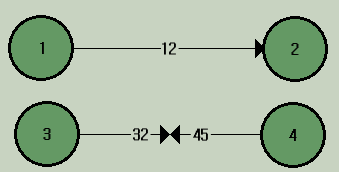
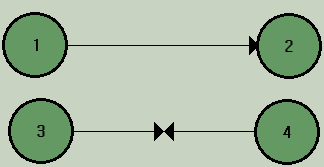




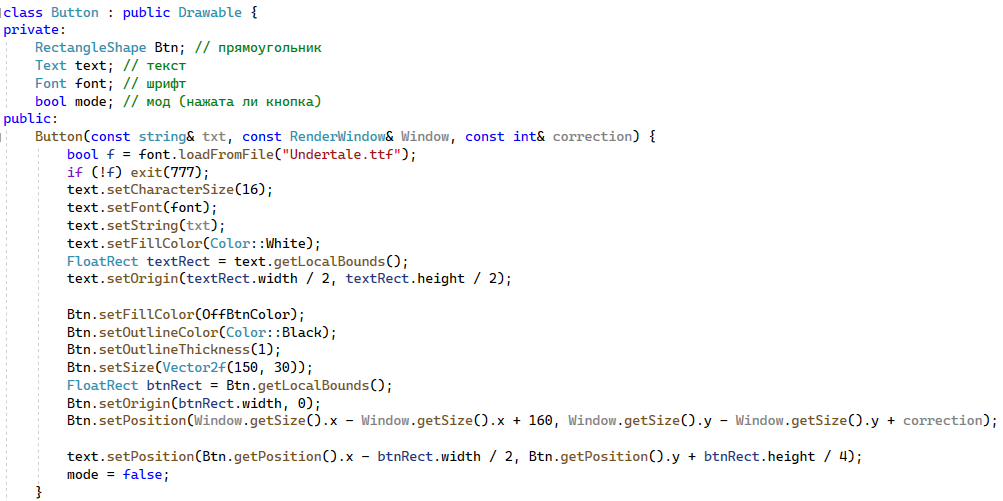
1. Класс EdgeLine - ребро, реализованное с помощью вершин Vertex и VertexCircle. Сам класс наследуем от класса Drawable, определенного в библиотеке SFML.

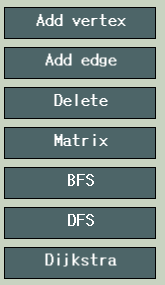
****

****

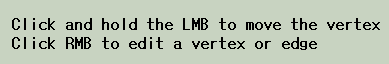
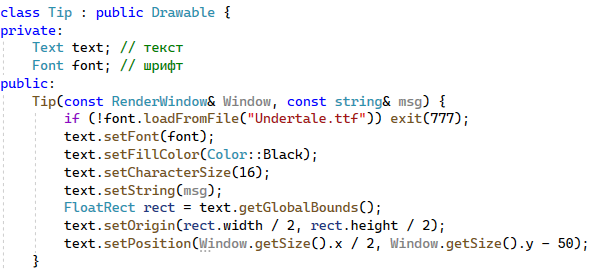
****

1. Класс Button - кнопка, реализованная с помощью прямоугольника. Имеет два режима - нажата или не нажата. Сам класс наследуем от класса Drawable, определенного в библиотеке SFML.



****

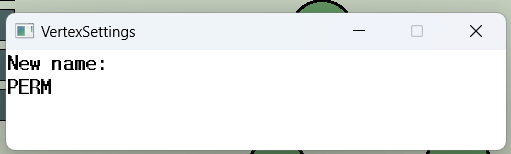
1. Класс Tip - вспомогательный класс, дающий пользователю подсказки о работе с интерфейсом программы, а также выдающий ответ на алгоритмы. Сам класс наследуем от класса Drawable, определенного в библиотеке SFML.



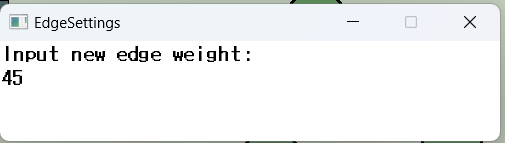
Сама программа является программой управляемой событиями. Она реагирует на следующие события:

1. Нажатие левой клавишей мыши по кнопкам и по вершинам;
2. Нажатие правой клавишей мыши:

* по вершинам - при нажатии появляется окно, в котором пользователь вводит новое имя для вершины;

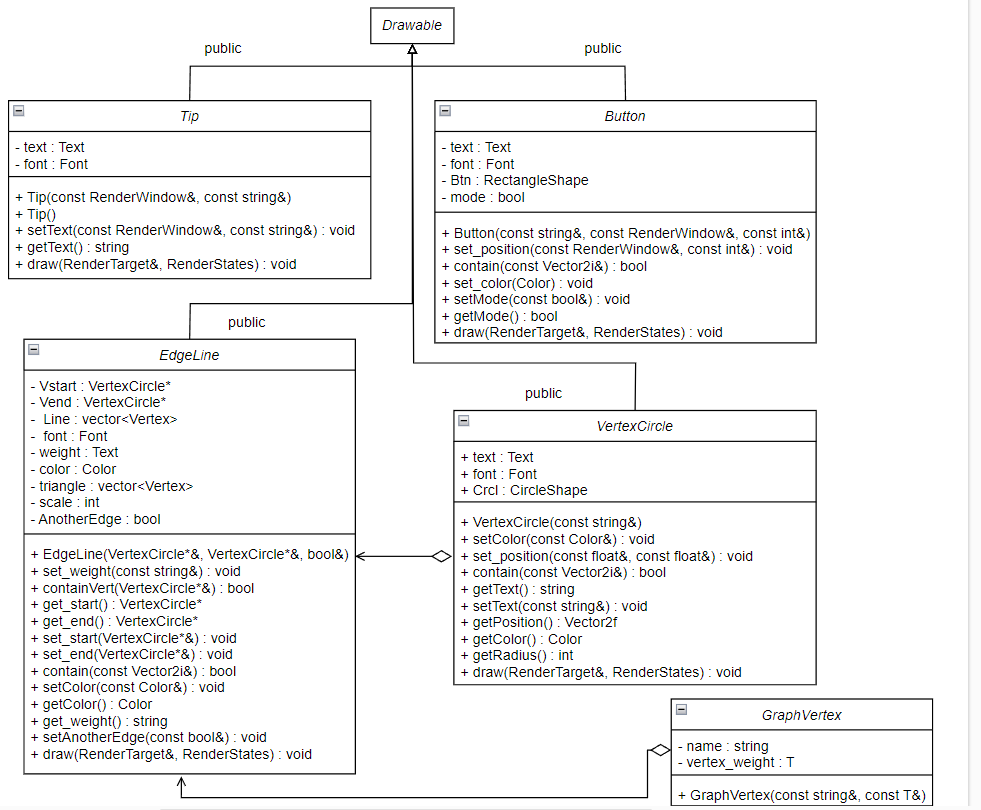


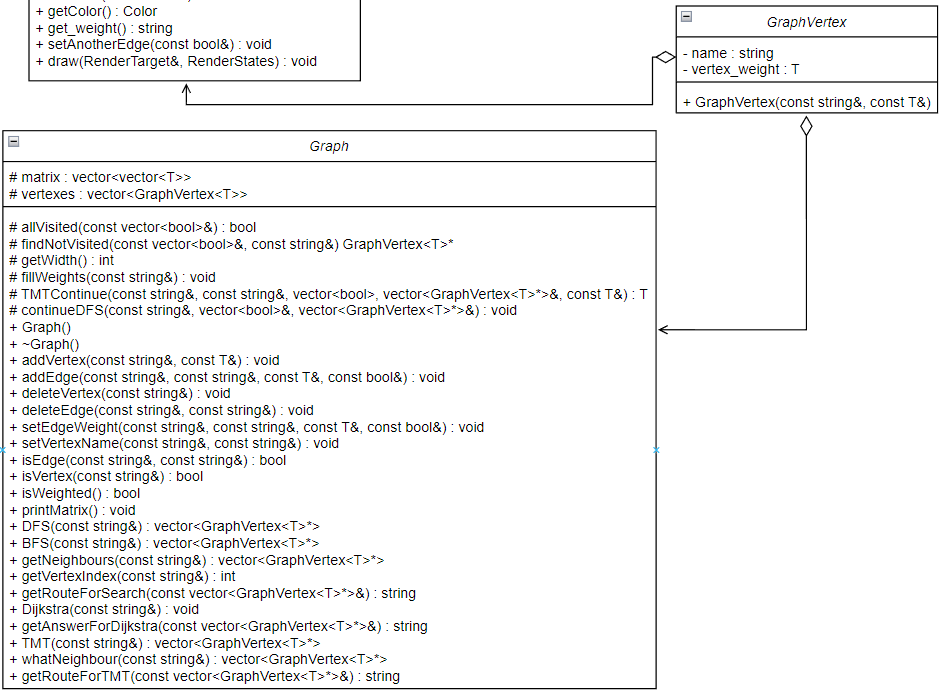
* по рёбрам - при нажатии появляется окно, в котором пользователь вводит вес ребра;



1. Движение мыши содержащей вершину с зажатой ЛКМ;
2. “Освобождение” кнопки мыши;
3. Изменение размеров окна.

# UML-диаграмма





# Работа программы

