Министерство науки и высшего образования Российской Федерации

Федеральное государственное автономное образовательное учреждение

высшего образования

**«Пермский национальный исследовательский**

**политехнический университет»**

Электротехнический факультет

Кафедра «Информационные технологии и автоматизированные системы»

направление подготовки: 09.03.01 - «Информатика и вычислительная техника»

**О Т Ч Е Т**

**по творческой работе**

**по дисциплине**

**«Основы алгоритмизации и программирования» (Семестр 2)**

Выполнил студент группы ИВТ-21-1б

Гребнев Алексей Дмитриевич

Проверил:

(оценка) (подпись)

г. Пермь-2022

**«Логарифмический калькулятор»**

**Постановка задачи:**

1. Реализовать калькулятор для вычисления логарифмов с разными основаниями
2. Оформить калькулятор
3. Проверить правильность выполнения действий

**Описание решения:**

Создание калькулятора выполнялось с помощью инструмента WinForms.

Для работы по реализации логарифмического калькулятора мне нужно было познакомится с WinForms( с её установкой, инструментарием и взаимодействию с ним). Трудности в работе возникли при выводе результата после вычисления результата в textbox. Данная проблема решилась после создания нового проекта, так как по комментариям в интернете люди часто сталкивались с этим.

В своей работе над логарифмическим калькулятором я горжусь тем, что пользователь может выполнить вычисление логарифма с полным удобством для себя. Он может очистить по своему желанию калькулятор от вписанных значений, заполнить кликом по кнопке случайными значениями и изучить вопрос «Что такое логарифм?». Также хочу отметить оформление калькулятора, в нем нет ничего лишнего, цвета подобраны хорошо и глазу приятно на это смотреть.

**Скриншоты калькулятора:**

****

Рисунок 16 (Вычисление двух логарифмов)



Рисунок 17 (Клик по кнопке "Полная очистка")

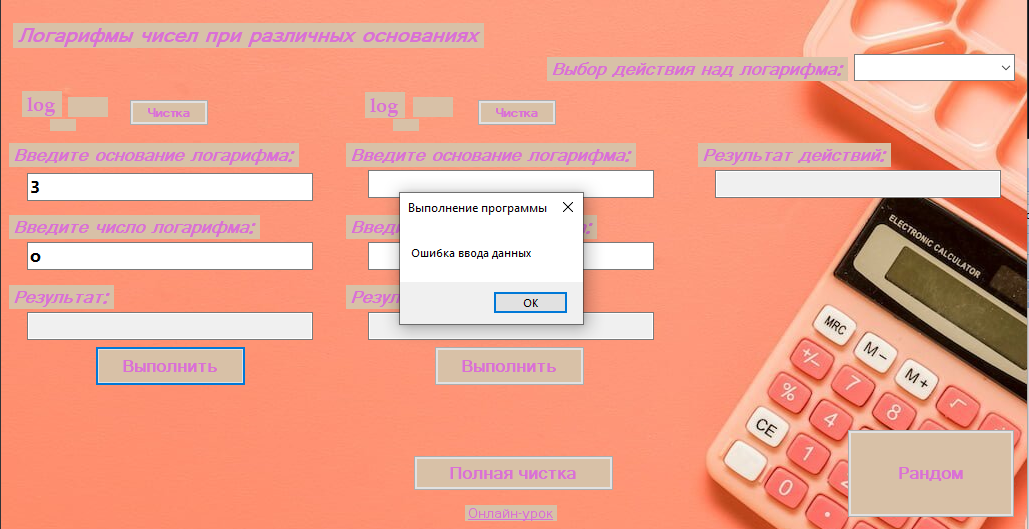


Рисунок 18 (Ошибка ввода значений логарифма)

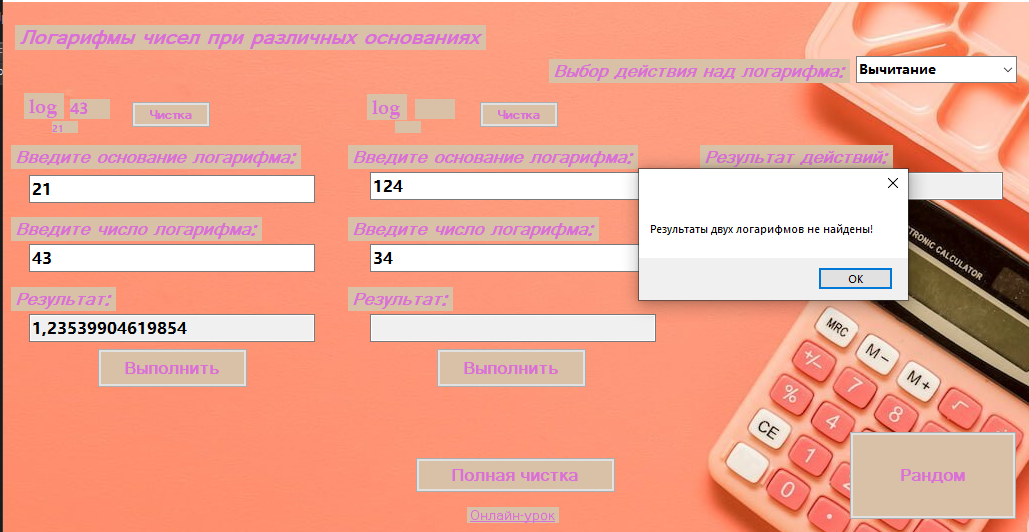


Рисунок 19 (Ошибка в выполнений действий над двумя логарифмами - отсутствует результат одного из логарифмов!)

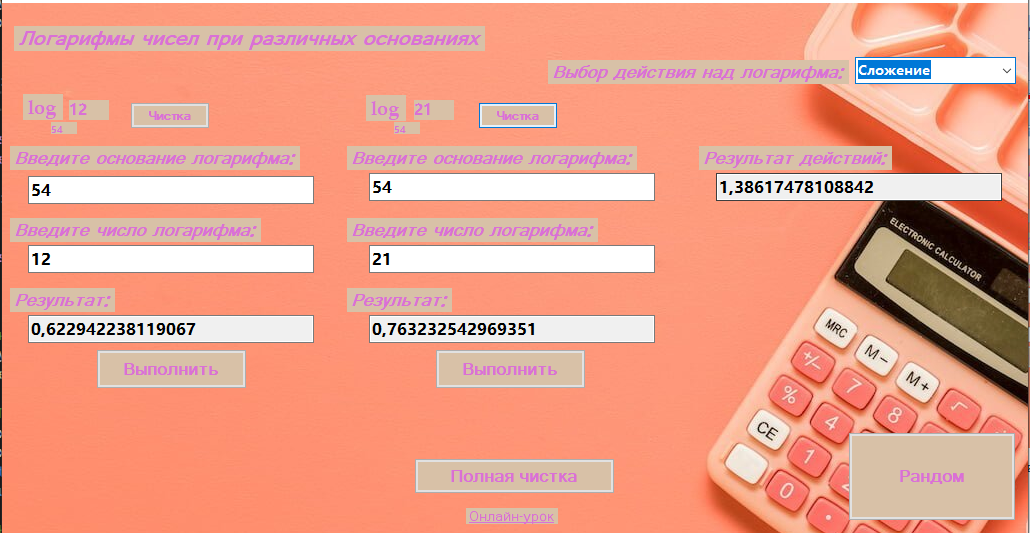


Рисунок 20 (Выполнение действий над логарифмами)



Рисунок 21 (Вывод в математическом виде записи)

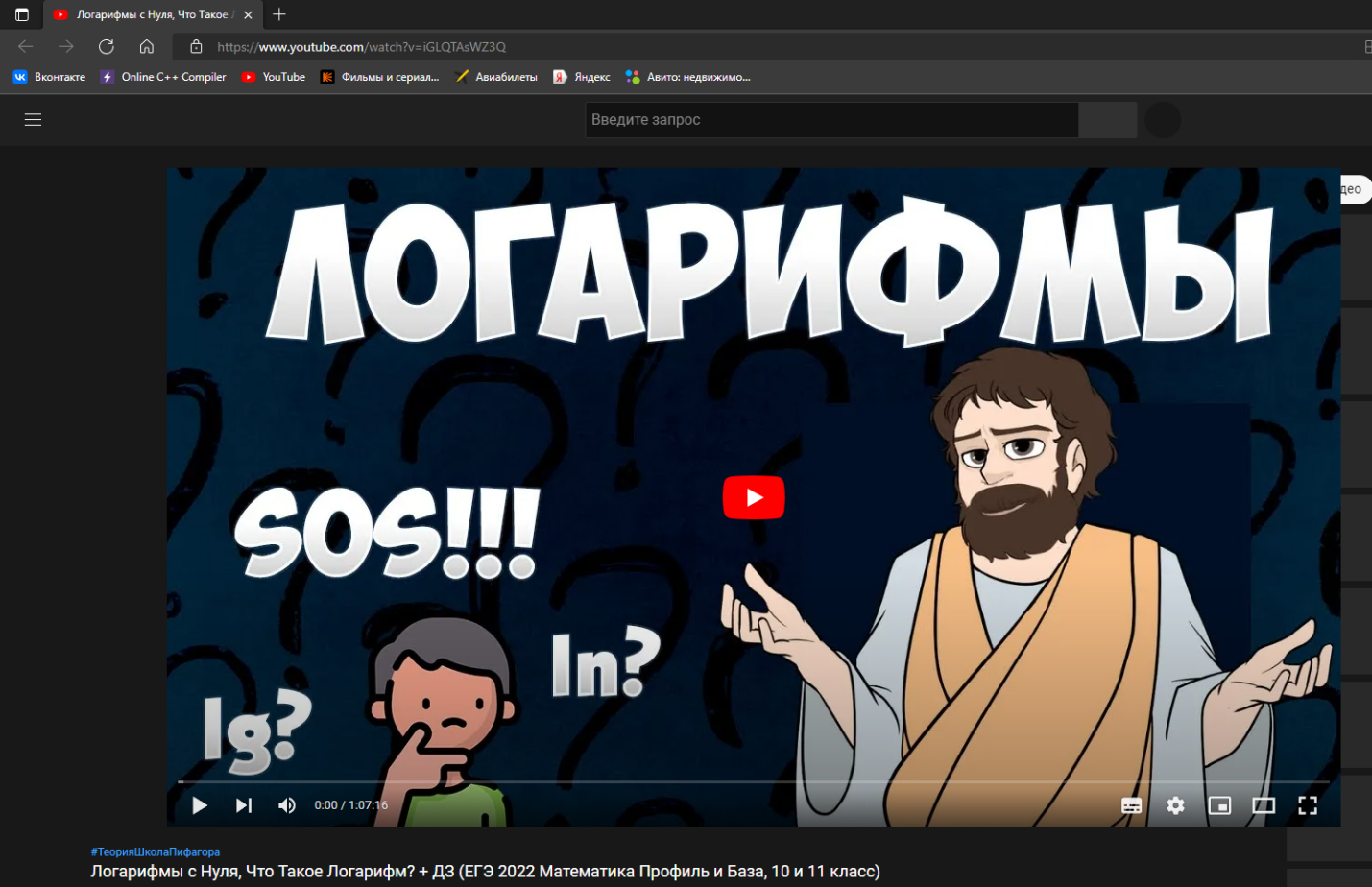


Рисунок 22 (Переход на онлайн-урок)

**Описание кода:**

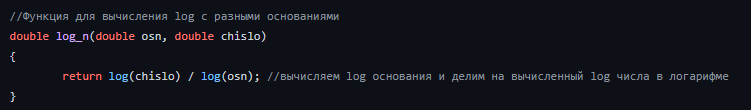


Рисунок 2 (Функция для вычисления логарифма)

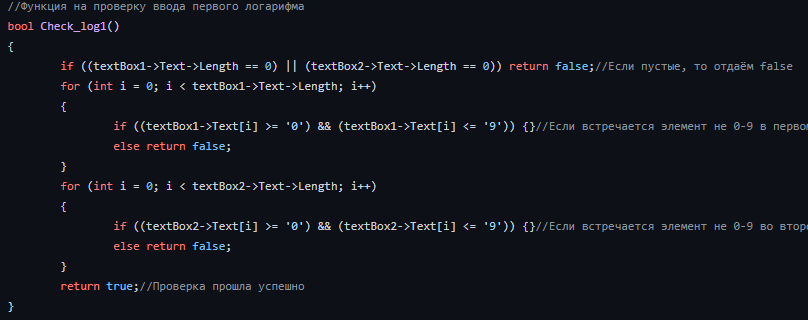


Рисунок 3 (Функция для проверки на ввод у 1-го логарифма)

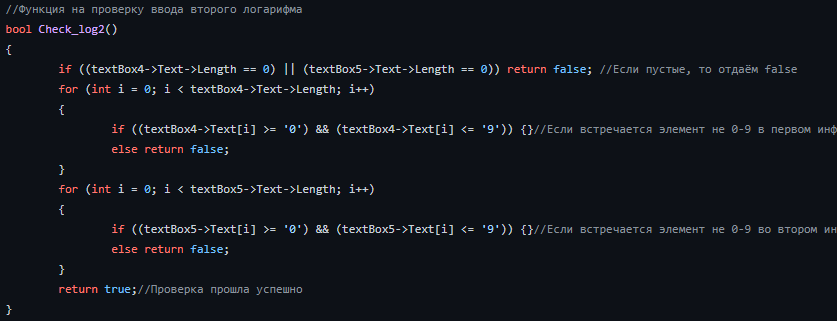


Рисунок 4 (Функция для проверки на ввод у 2-го логарифма)

Две этих функции на проверку ввода возвращают bool значение по условиям:  
1)Если отсутствует(длина текста в textbox будет равняться нулю) хотя бы одно значение для вычисления логарифма (число и основание), то возвращает false  
2) Если встречается элемент не 0-9 в первом textbox, то возвращаем false  
3) Если встречается элемент не 0-9 во втором textbox, то возвращаем false  
4)Если не одно из этих условий не дошло до возврата, то возвращаем true

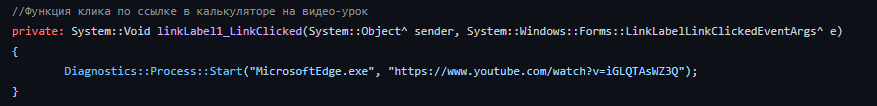


Рисунок 5 (Клик по ссылке в калькуляторе)

Ссылка на онлайн-урок по теме «Логарифм с нуля»

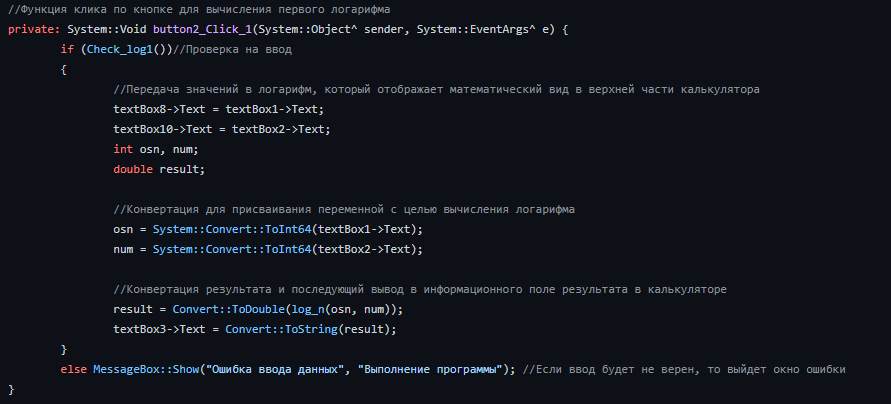


Рисунок 6 (Кнопка для вычисления первого логарифма)

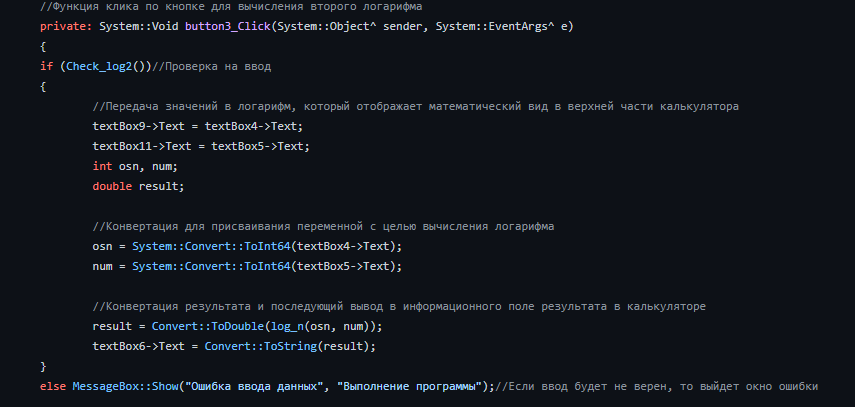


Рисунок 7 (Кнопка для вычисления второго логарифма)

В двух этих кнопках-подсчётов мы используем:  
1)Предварительную проверку на ввод значений логарифма (Но! Если проверка не пройдена то калькулятор выводит окно ошибки «Ошибка ввода данных»);  
2)Функцию для вычисления логарифмов (при конвертированных значениях, взятых из textbox значений логарифма);  
3)Передачу результата в textbox результата логарифма;

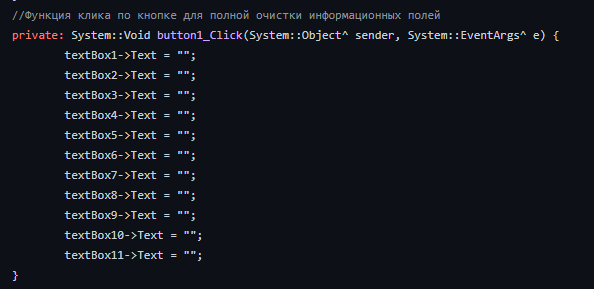


Рисунок 8(Кнопка полной чистки калькулятора от вводимых значений)

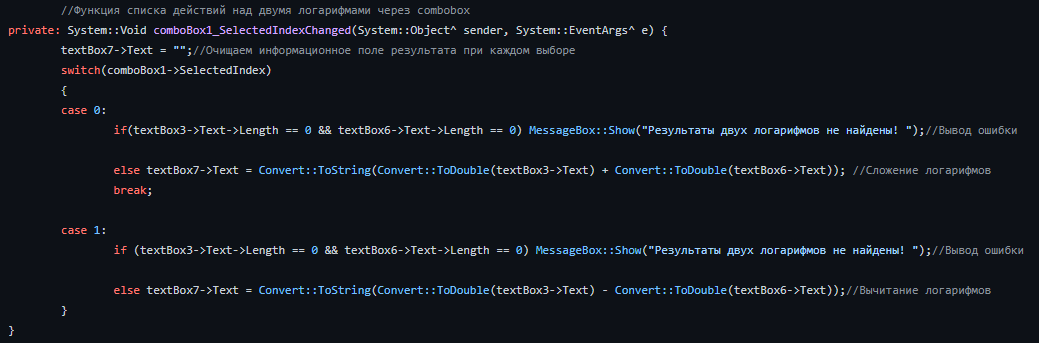


Рисунок 9(Функция списка действий над двумя логарифмами через combobox)

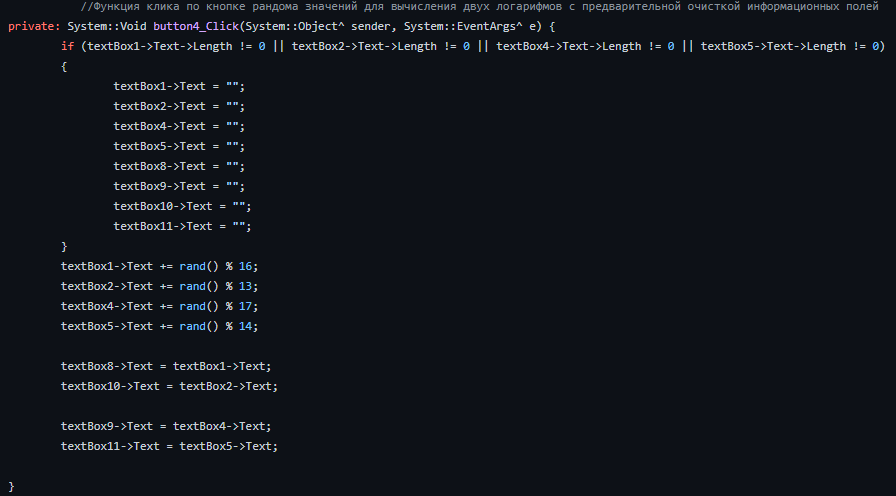


Рисунок 10 (Кнопка записи случайных значений для двух логарифмов)

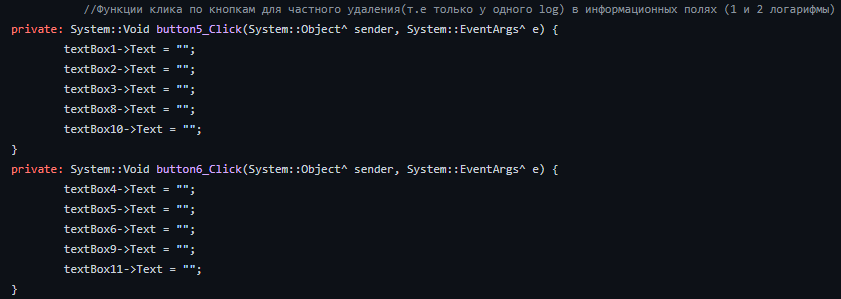


Рисунок 11 (Две кнопки для частной чистки значений в 1-ом и 2-ом логарифме)

**UML диаграмма:**

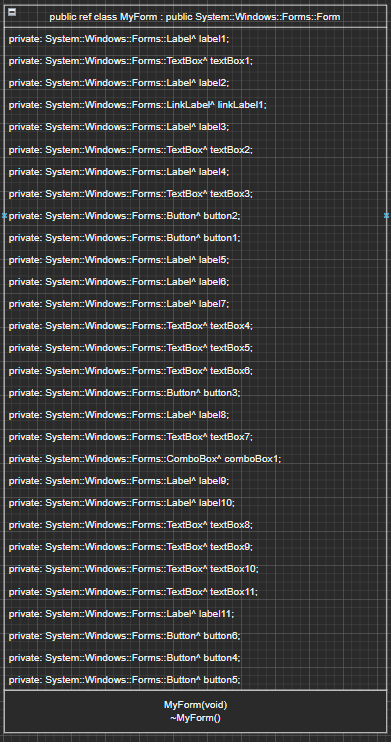
****

Рисунок 22

**«Задача Коммивояжера»**

**Постановка задачи:**

1. Реализовать функции для решения задачи методом ветвей и границ;
2. Реализовать нахождение кратчайшего пути, используя реализованные функции в пункте (1)
3. Вывести кратчайший путь и сумма пройденного расстояния
4. Отрисовка графа с помощью SFML

**Описание решения:**

Создание программы реализовывалось на основе метода ветвей и границ, для отрисовки графа был использован SFML(базовый инструментарий был изучен для работы с данным заданием). Трудность возникала с позиционированием ребра к вершинам, так как простановка координат концов рёбер случайным способом для получения желаемого результата – довольно муторно. Решение: классы и математический подход в функции этого класса для вычисления угла для поворота, чтобы вершина соединяла центры двух окружностей.

В задаче я горжусь понятной и приятной визуализации графа, отображения кратчайшего пути и суммы пути их веса, а также аккуратного для понимания оформления кода программы.

**Скриншоты программы с Коммивояжером:**

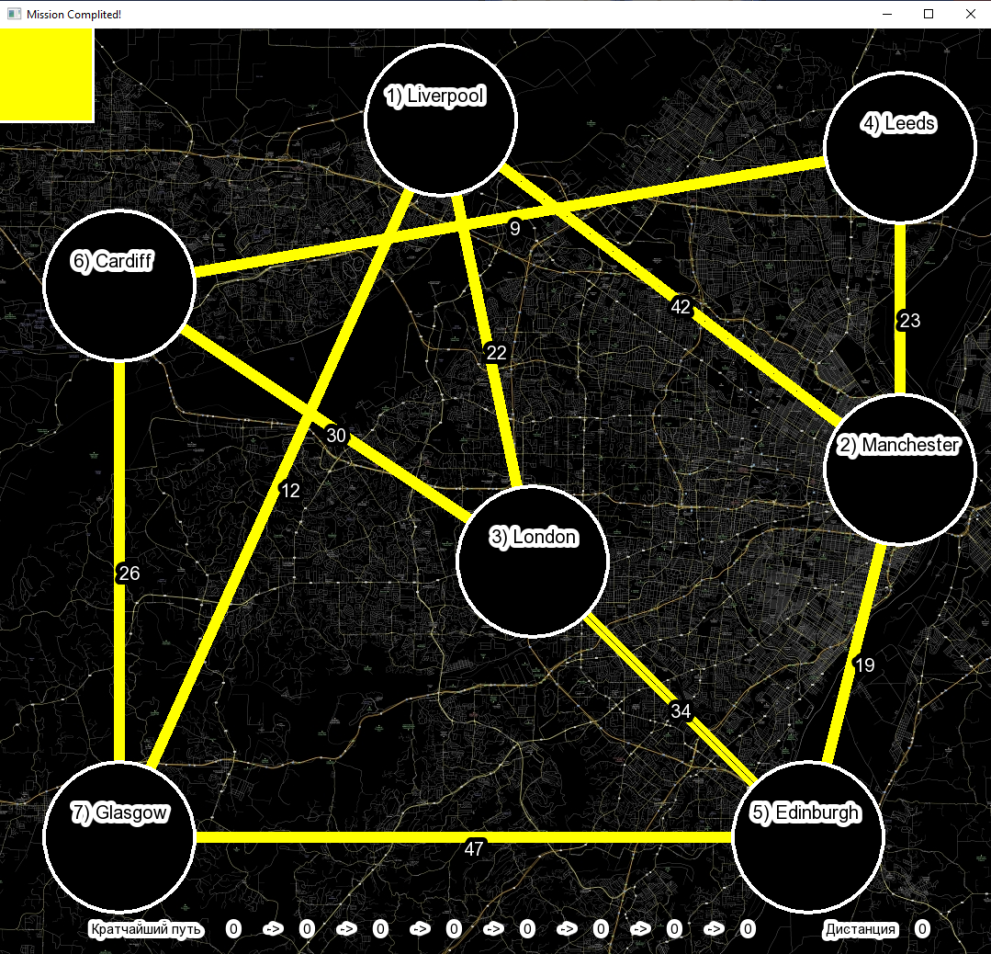


Рисунок 23 (Вывод графа)

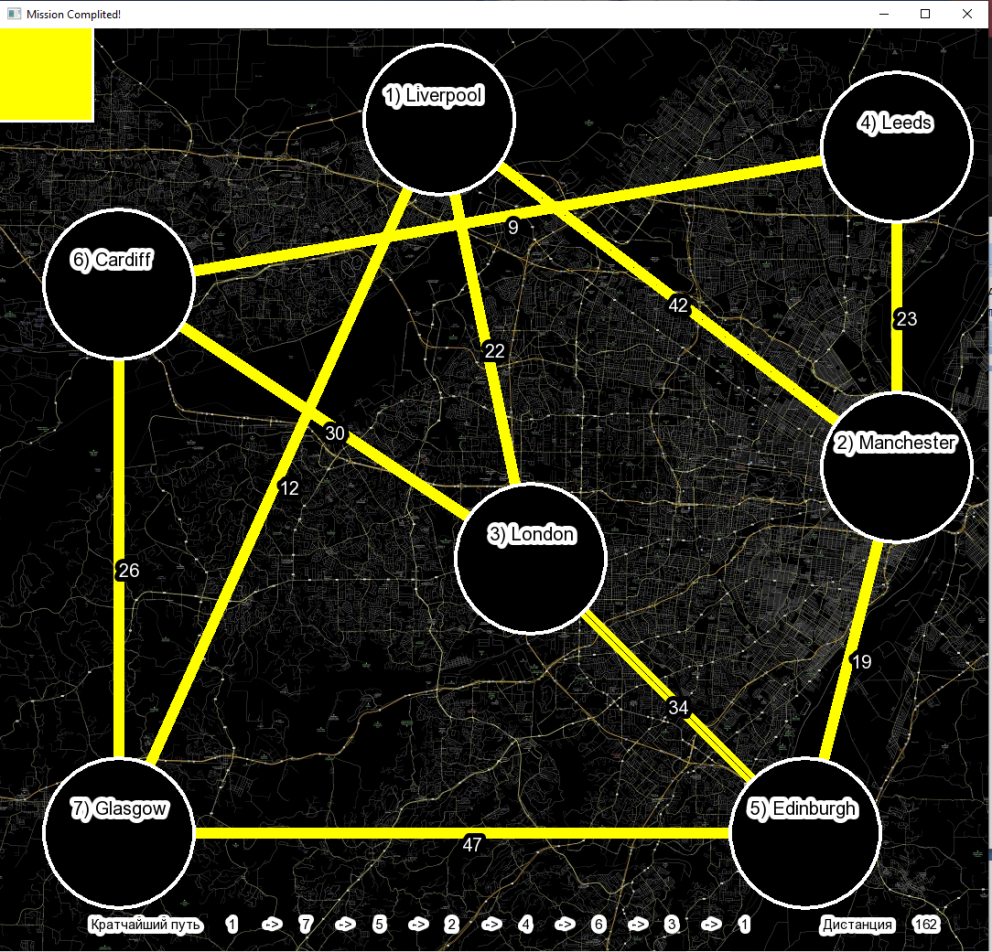


Рисунок 24 (Вычисление и вывод результата нахождения кратчайшего пути и суммы веса его путей)

**Описание кода:**



Рисунок 25

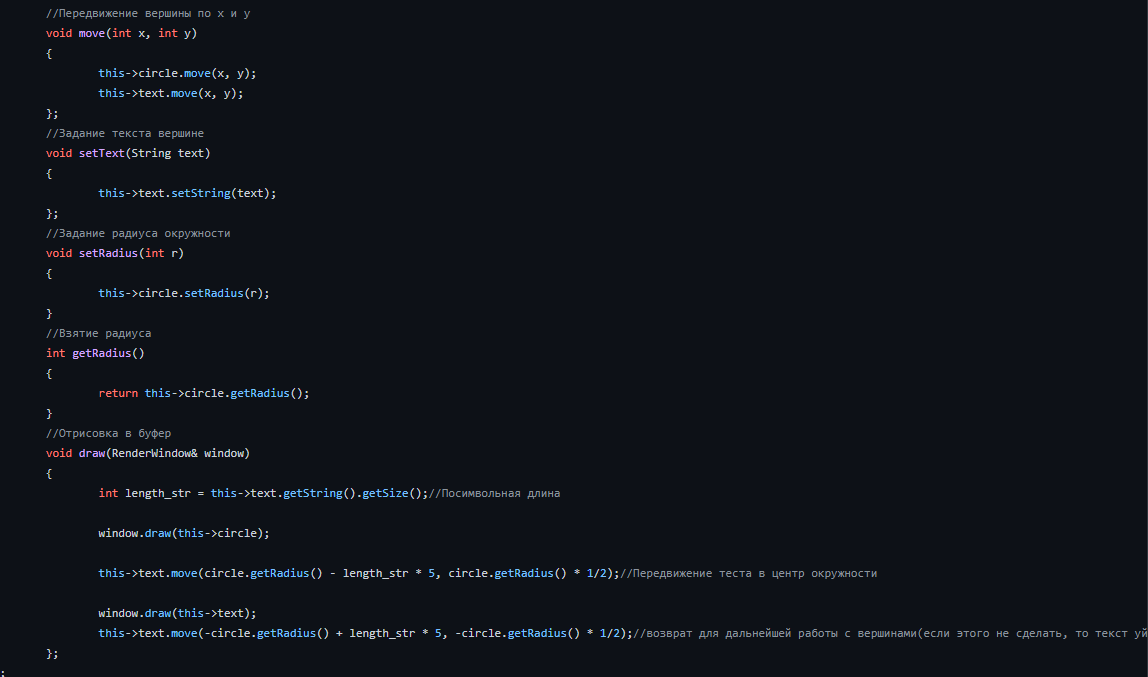


Рисунок 26

**Примечание:** рис. 25 и 26 это реализация класса вершины(окружность и текст внутри)

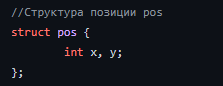
****

Рисунок 27 (структура позиции по x и y координатам)



Рисунок 28

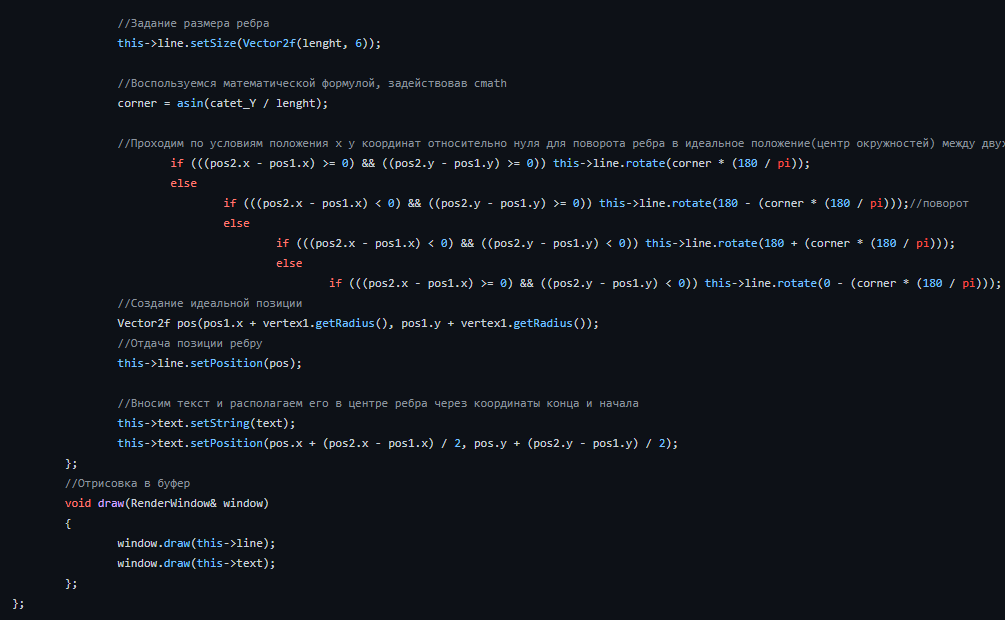


Рисунок 29

**Примечание:** рис. 28 и 29 это реализация класса ребра(линия и текст-вес посередине линии )

Линия получает позицию через функцию setVertex, которая математическим подходом вычислений поворачивает ребро(rotate) относительно двух соединяемых вершин, а точнее их центров.

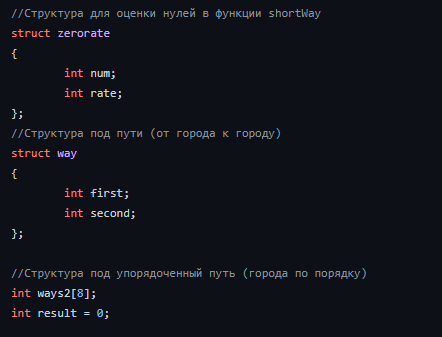


Рисунок 30 (Структуры для работы в функции нахождения кратчайшего пути)

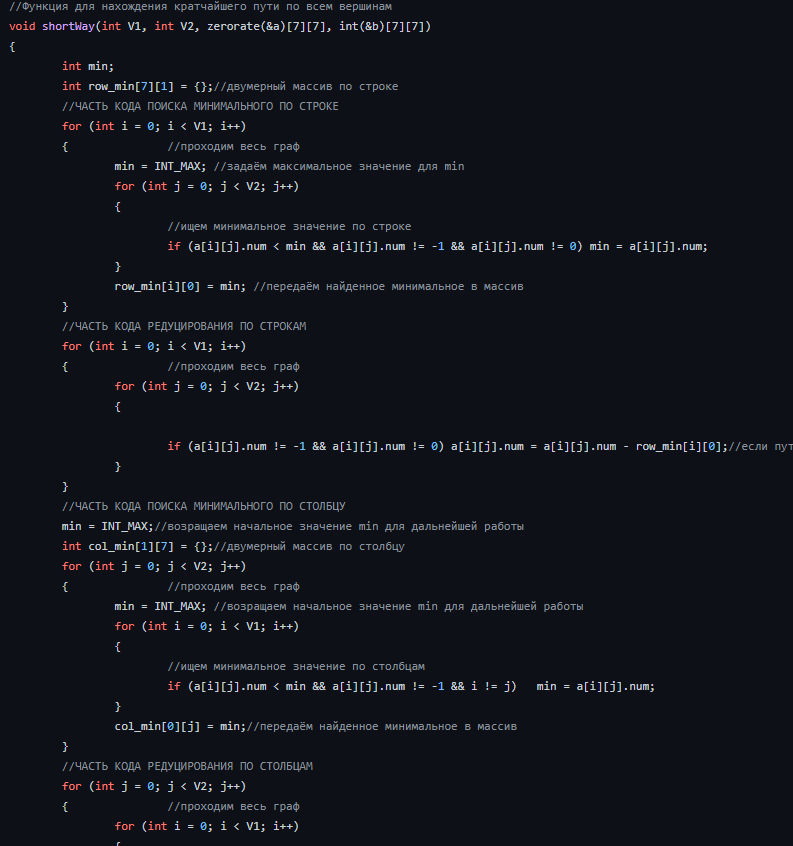


Рисунок 31



Рисунок 32

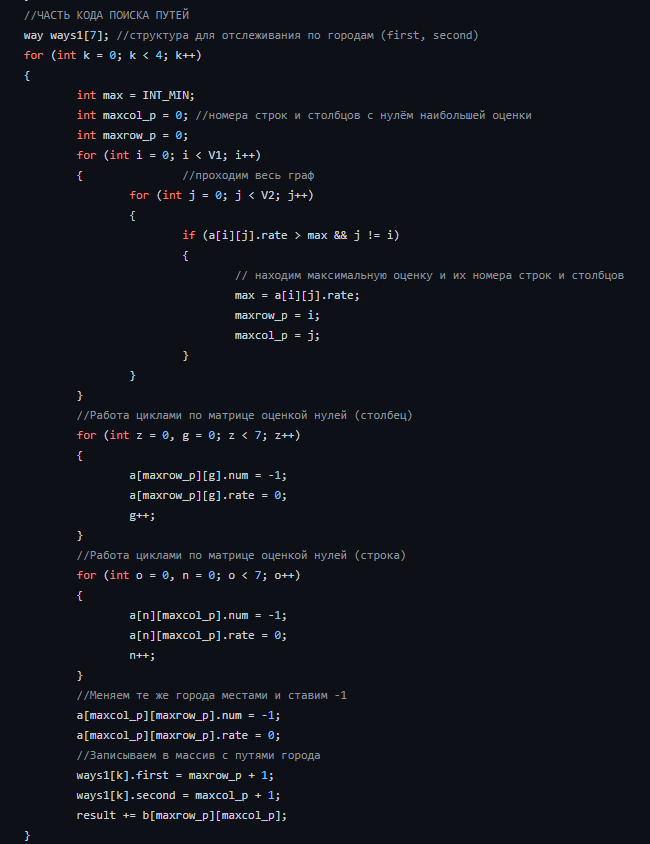


Рисунок 33

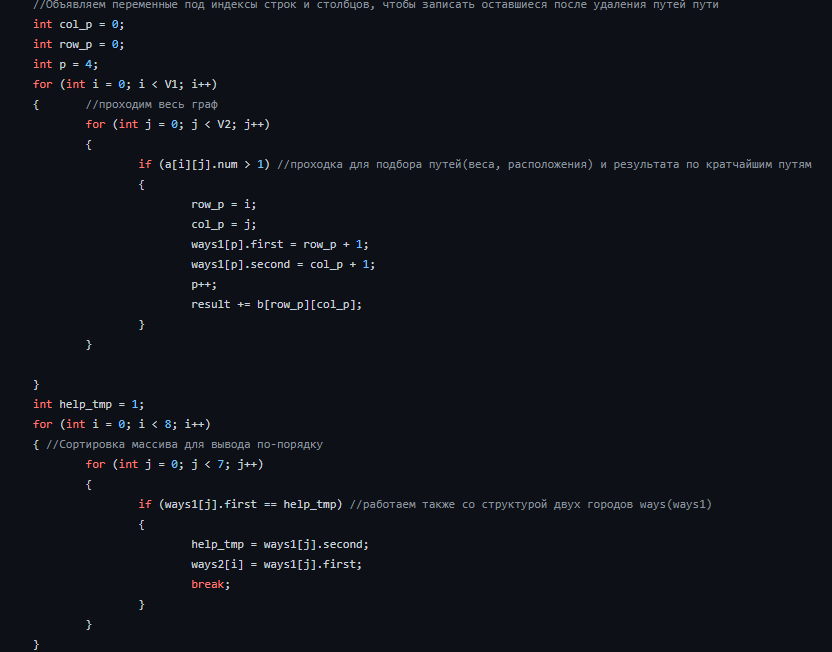


Рисунок 34

**Примечание:** рис. 30-34 это функция shortWay, которая находит кратчайший путь с помощью матрицы смежности и оценки нулей(с помощью структуры zerorate)

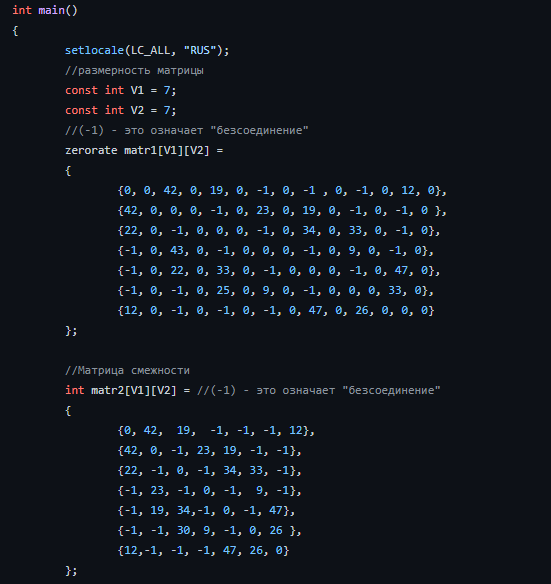


Рисунок 35 (создание двух двумерных массивов для оценки нулей и для матрицы смежности)

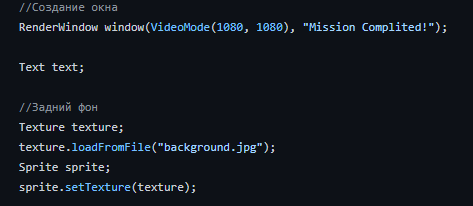


Рисунок 36 (Создаём окно, текст и задний фон окна)



Рисунок 37 (Создание и заполнение вершин информацией)

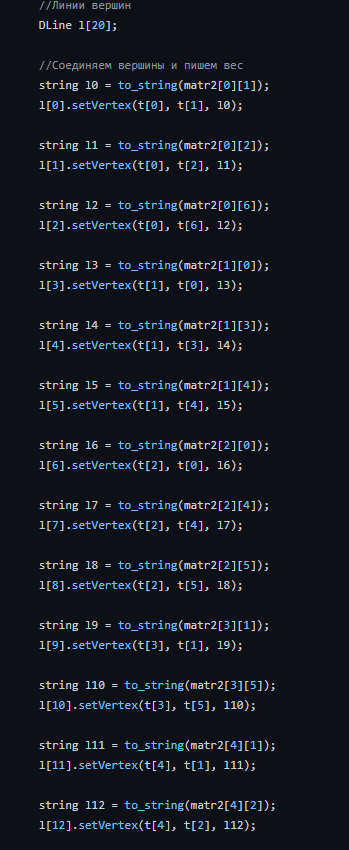


Рисунок 38 (Создание и заполнение рёбер информацией)

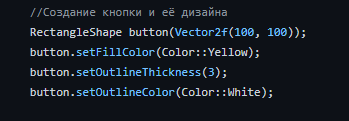


Рисунок 39 (Создание кнопки в левом верхнем углу)

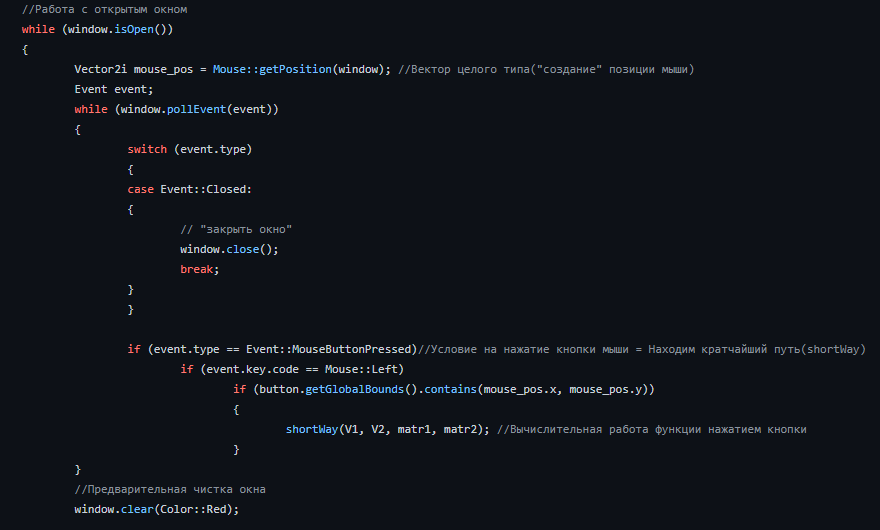


Рисунок 40 (Создание условия открытого окна и нажатия кнопки в нём)

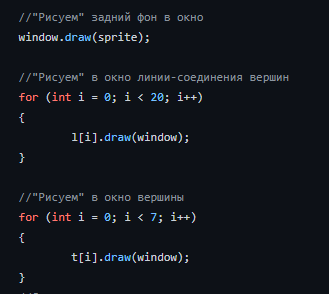


Рисунок 41

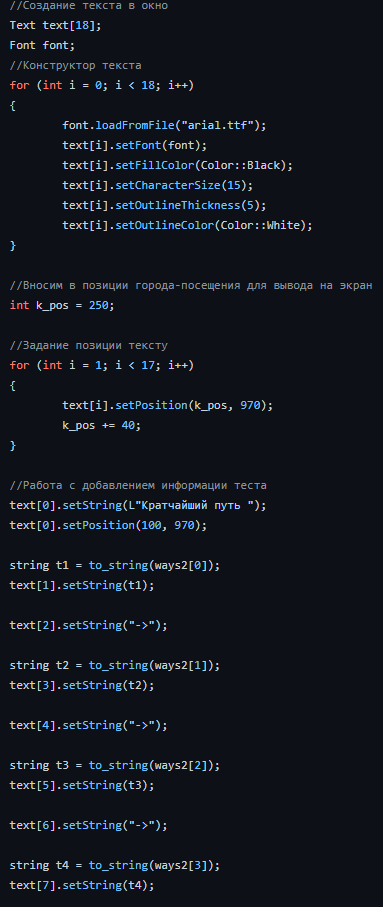


Рисунок 42 (Создание текста для отрисовки найденного кратчайшего пути в окне)

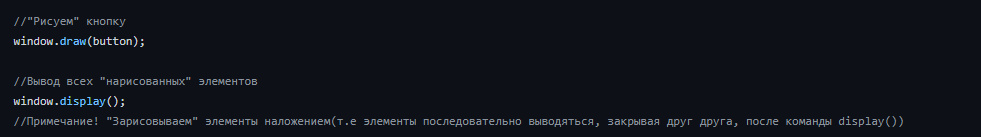


Рисунок 43(Отрисовка кнопки и ФИНАЛЬНЫЙ вывод в окно из буфера, который мы составили последовательно)

**UML диаграмма:**

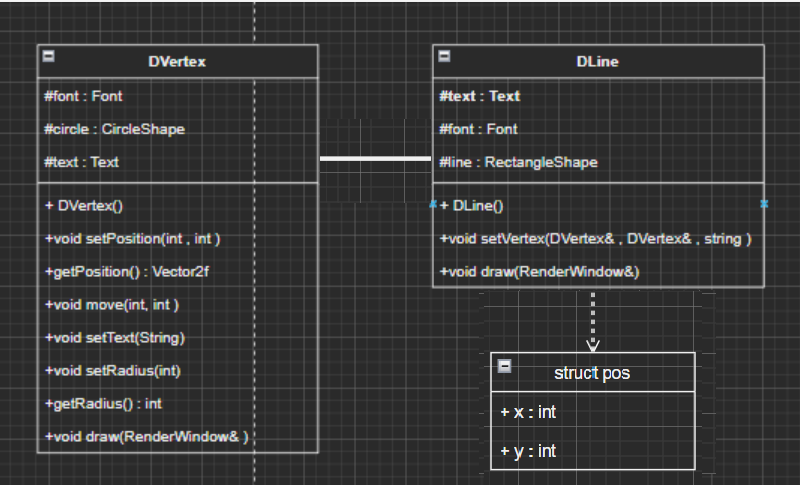


Рисунок 44