Министерство образования и науки Российской Федерации  
Федеральное‌ ‌государственное‌ ‌бюджетное‌ ‌образовательное‌ ‌учреждение‌

высшего‌ ‌образования‌

**«Пермский национальный исследовательский  
политехнический университет»**

Кафедра «Информационные технологии и автоматизированные системы»

**ОТЧЁТ**

**по творческому заданию**

Тема: Разработка калькулятора и задача Коммивояжёра

Выполнил:

Студент группы РИС-20-1б

Ознобихин Елисей Андреевич

Проверила:

Доцент кафедры ИТАС

Полякова О. А.

**Пермь, 2021**

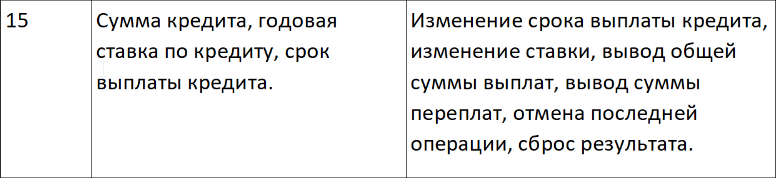
**Калькулятор**

**Цель работы**

Разработать универсальный калькулятор для работы с кредитами.

**Постановка задачи**

1. Разработать алгоритм калькулятора в соответствии с заданием.
2. Реализовать алгоритм в виде программы на алгоритмическом языке С++.
3. Разработать интерфейс средствами Qt.
4. Использование анимации, автоматизации, креативной визуализации повышает рейтинг автора проекта и влияет на экзаменационную оценку.

****

**Анализ задачи**

1. Приложение построено на следующем алгоритме:
   1. При открытии окна пользователю необходимо выбрать одно из трех полей ввода. Выбор поля осуществляется путем нажатия на него левой кнопкой мыши.
   2. Используя специальные кнопки калькулятора “0” – “9”, “.” (знак для выделения дробной части) и “**←**” (удаление крайнего символа), пользователь изменяет поля ввода по собственному желанию. Изменение поля реализовано в следующем виде: происходит считывание текущей строки поля ввода, изменение считанной строки, исходя из нажатой кнопки. В поле ввода записывается измененная строка.
   3. Пользователю представлена возможность полного очищения всех вводимых / выводимых полей при помощи кнопки “Сброс результата!”. Удаление содержимого всех полей выполнено путём вставки специального текста, указывающего на отсутствие введенных / выведенных символов.
   4. Для подсчета выплат и переплат и демонстрации результатов вычислений в полях вывода пользователь должен выбрать способ подсчета выплат и переплат по кредиту и произвести нажатие на кнопку “Посчитать!”. Выбор способа подсчета осуществляется нажатием кнопки “ Аннуитетный платеж ” или “Дифференцированный платеж”, по умолчанию способ не выбран. Реализация вычисления представлена следующим образом: по нажатию кнопки происходит считывание всех полей ввода и работа по алгоритму вычисления всех необходимых параметров, используя специальные формулы для кредитов.
2. Структура программы представлена следующим образом:

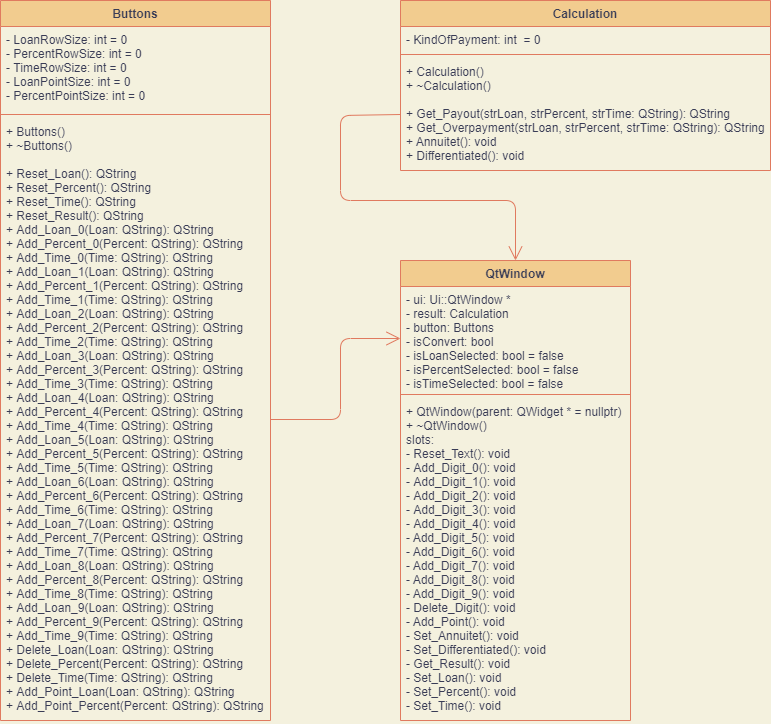
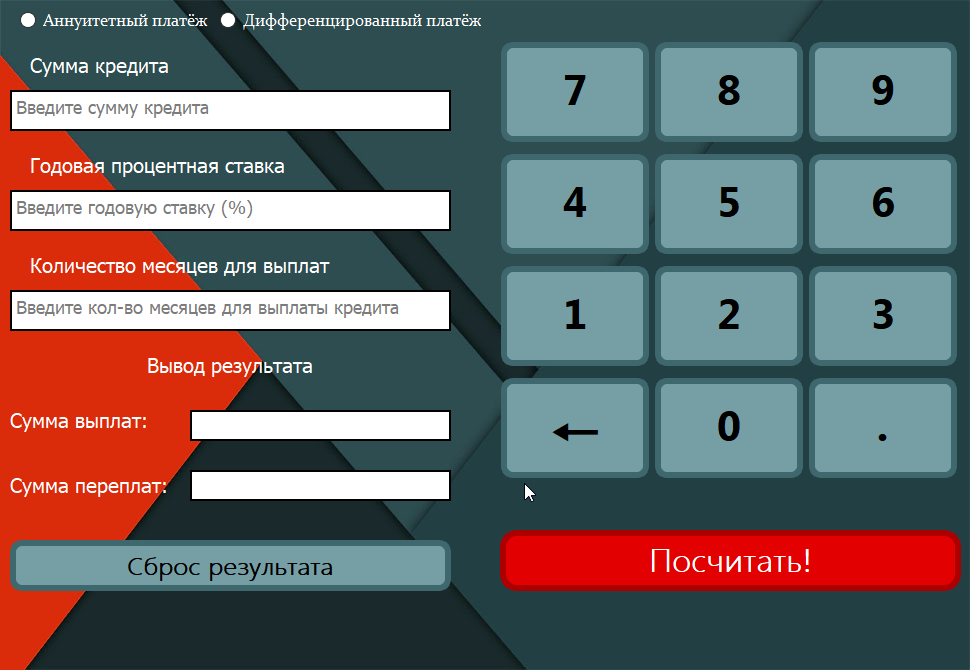


Рис. 1. UML-диаграмма калькулятора

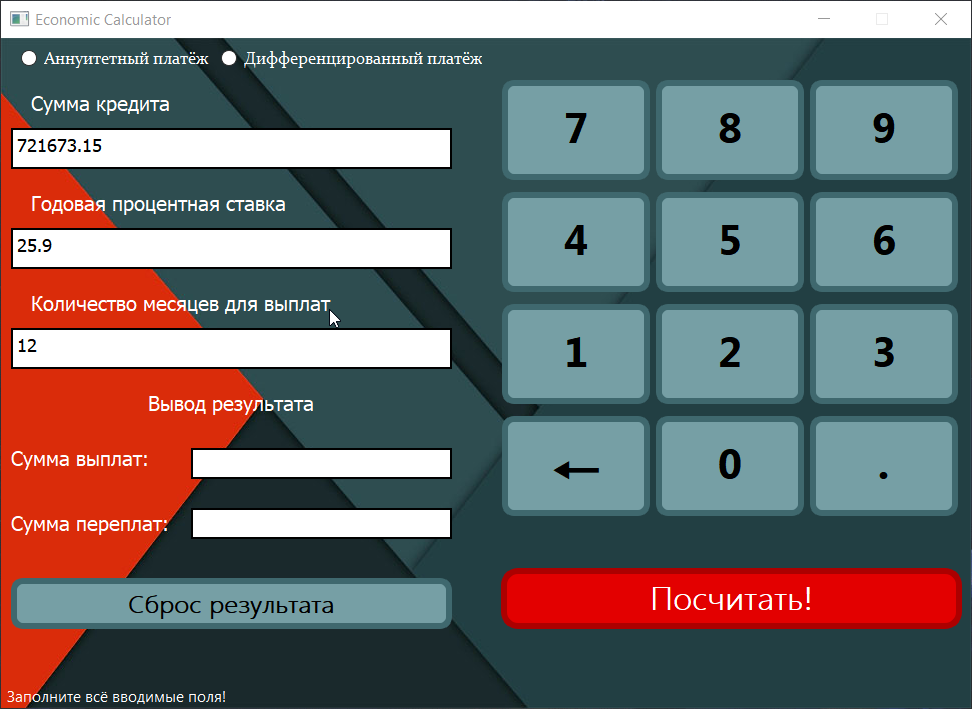
* 1. Класс QtWindow – главное окно программы. Отвечает за реализацию нажатий кнопок и полей ввода / вывода для взаимодействия программы с пользователем. Имеет связь ассоциация с классами Buttons и Calculation.
  2. Класс Buttons – кнопки для изменения полей ввода / вывода (кнопки “0” – “9”, “.” (знак для выделения дробной части), “**←**” (удаление крайнего символа), “Сброс результата”). Отвечает за работу ввода и удаления строк из полей ввода. Имеет связь ассоциация с классом QtWindow.
  3. Класс Calculation – кнопка подсчёта и вывода результата. Отвечает за реализацию вычисления и вывода сумм выплат и переплат. Имеет связь ассоциация с классом QtWindow.

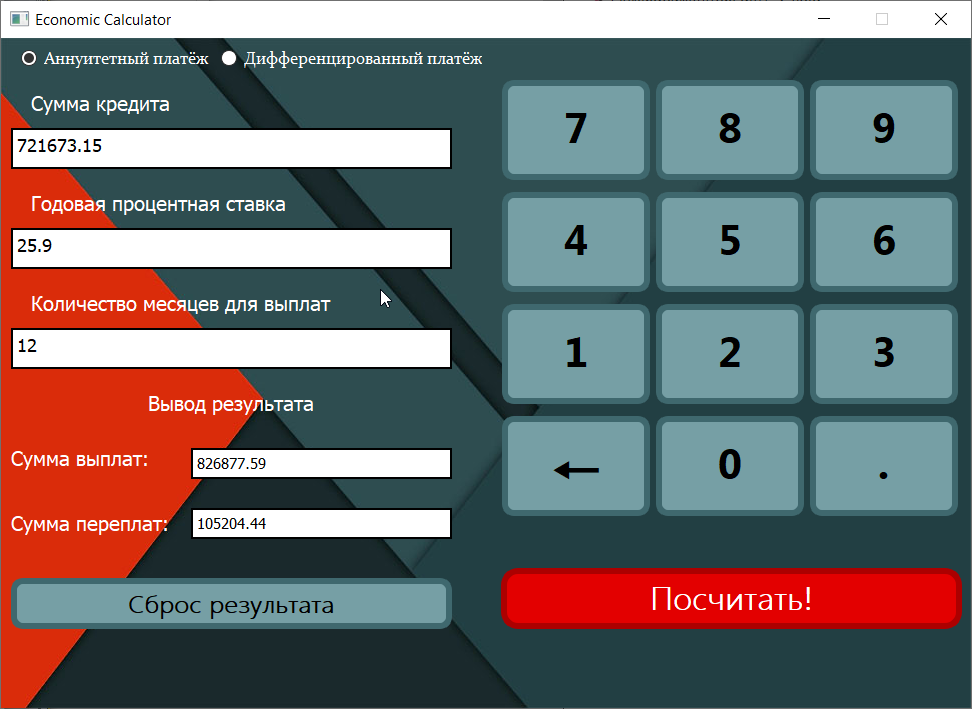
**Тестирование работы программы**

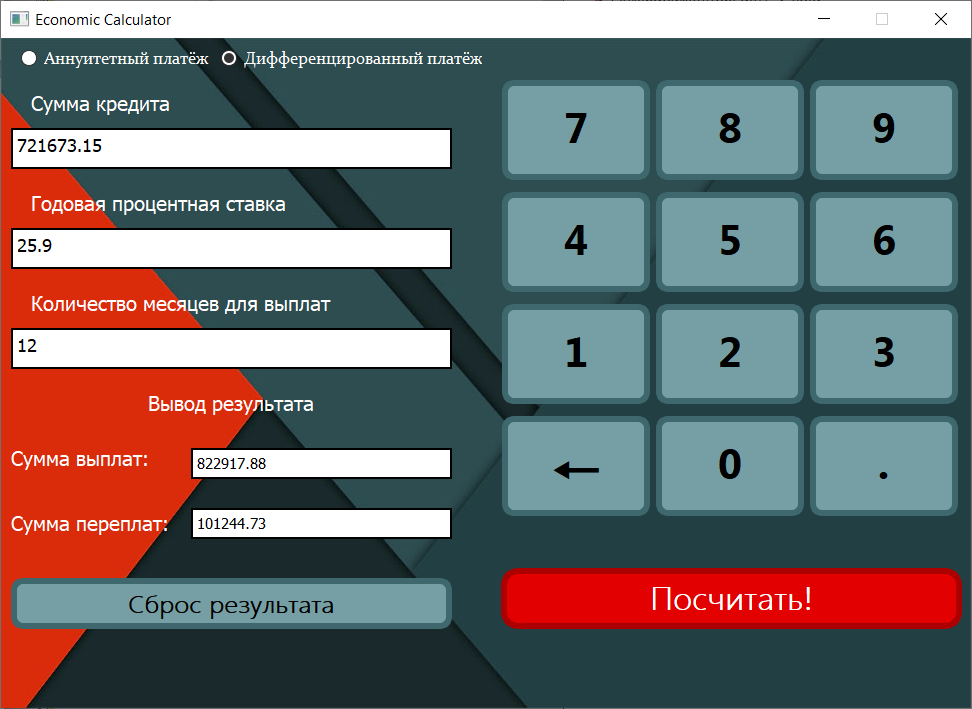
Ниже приведены скриншоты работы программы:











**Задача Коммивояжёра**

**Цель работы**

Разработать универсальный калькулятор для работы с кредитами.

**Постановка задачи**

1. Реализовать визуализацию графа в Qt или OpenGL.
2. Разработать интерфейс средствами Qt или OpenGL.
3. Реализовать алгоритм решения задачи Коммивояжёра для графа.
4. Использование анимации, автоматизации, креативной визуализации повышает рейтинг автора проекта и влияет на экзаменационную оценку.

**Анализ задачи**

1. Приложение построено на следующем алгоритме:
   1. В главном окне пользователю доступны визуализированный граф, одно поле для ввода вершины для решения задачи Коммивояжёра, два поля для вывода маршрута и длины пройденного пути и пять кнопок для взаимодействия пользователя с программой. Выбор поля осуществляется путем нажатия на него левой кнопкой мыши.
   2. При нажатии кнопки “Создать!” пользователю будет предложено ввести количество узлов графа и матрицу смежности для построения графа. Ввод осуществляется через консольное приложение.
   3. При нажатии кнопки “Добавить!” создается новое диалоговое окно для изменения дорог между узлами графа. Пользователю необходимо правильно заполнить все поля ввода для корректного построения нового узла графа с одной дорогой и нажать на кнопку “Add Node”. Если граф состоит из нескольких узлов и необходимо соединить один узел с несколькими другими, пользователю необходимо правильно заполнить все поля ввода для корректного построения нового узла графа с одной дорогой и нажать на кнопку “Add Road”.
   4. При нажатии кнопки “Изменить!” создается новое диалоговое окно для добавления узла графа. Пользователю необходимо правильно заполнить все поля ввода для корректного построения новой дороги и нажать на кнопку “Edit Node”. Если граф состоит из нескольких узлов и необходимо изменить несколько дорог, пользователю необходимо правильно заполнить все поля ввода для корректного построения новой дороги и нажать на кнопку “Edit Road”.
   5. При нажатии кнопки “Удалить!” создается новое диалоговое окно для удаления узла графа. Пользователю необходимо правильно заполнить поле ввода для удаления узла и нажать на кнопку “Delete Node”.
   6. Пользователю необходимо ввести номер начальной вершины, с которой будет решаться задача Коммивояжёра, в поле “Начальная вершина”.
   7. При нажатии кнопки “Маршрут” происходит решение задачи Коммивояжёра методом ветвей и границ, построение маршрута, начиная с номера узла, введённого в поле ввода “Начальная вершина” с последующим выводом в поле “Маршрут” и вычисление длины найденного маршрута и вывод найденной длины в поле “Длина маршрута”.
2. Структура программы представлена следующим образом:

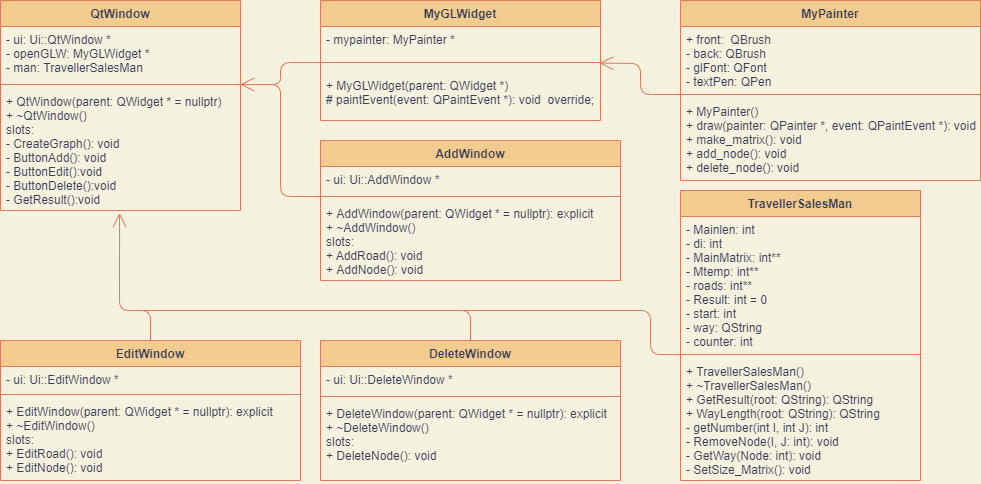
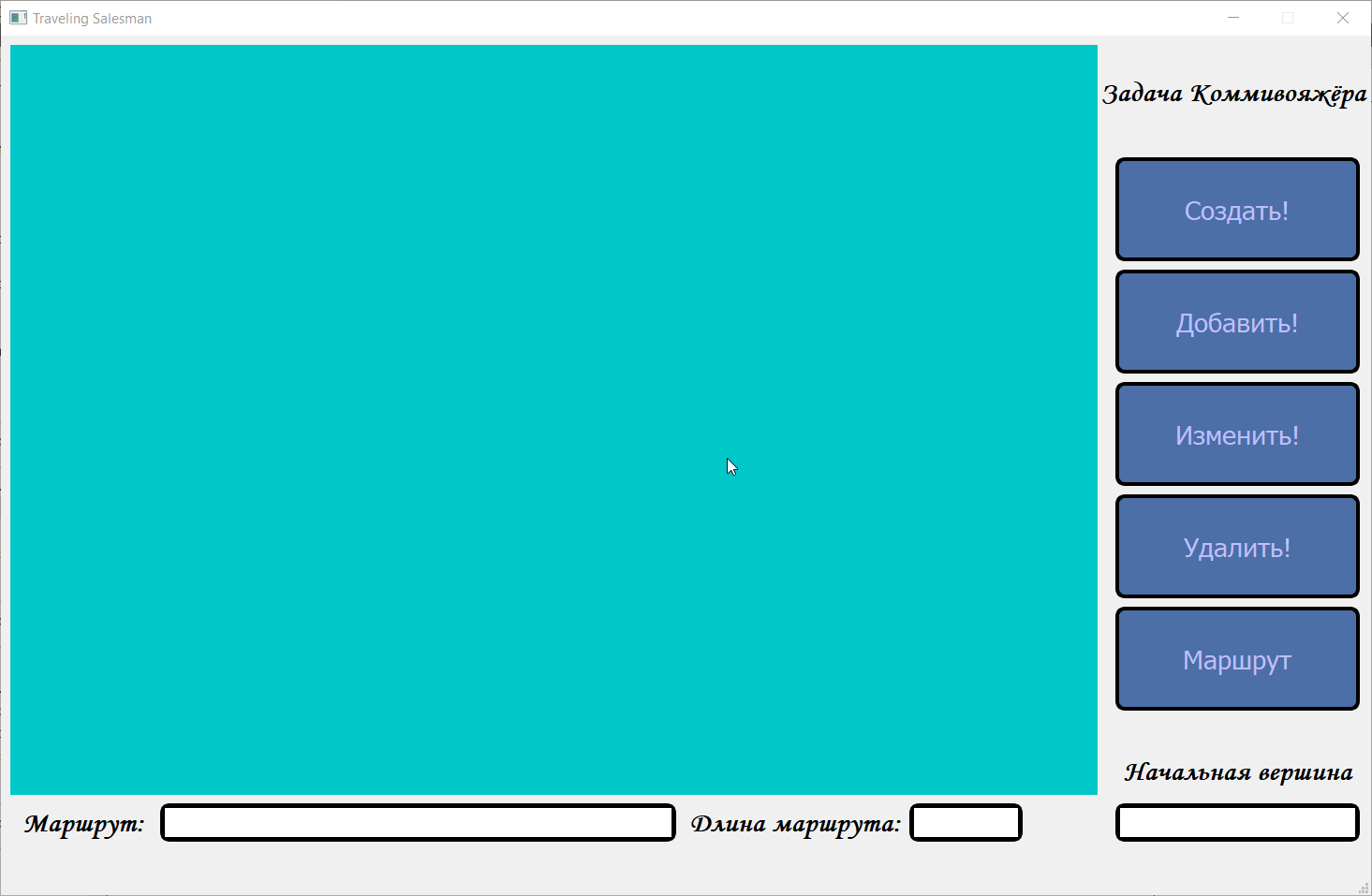


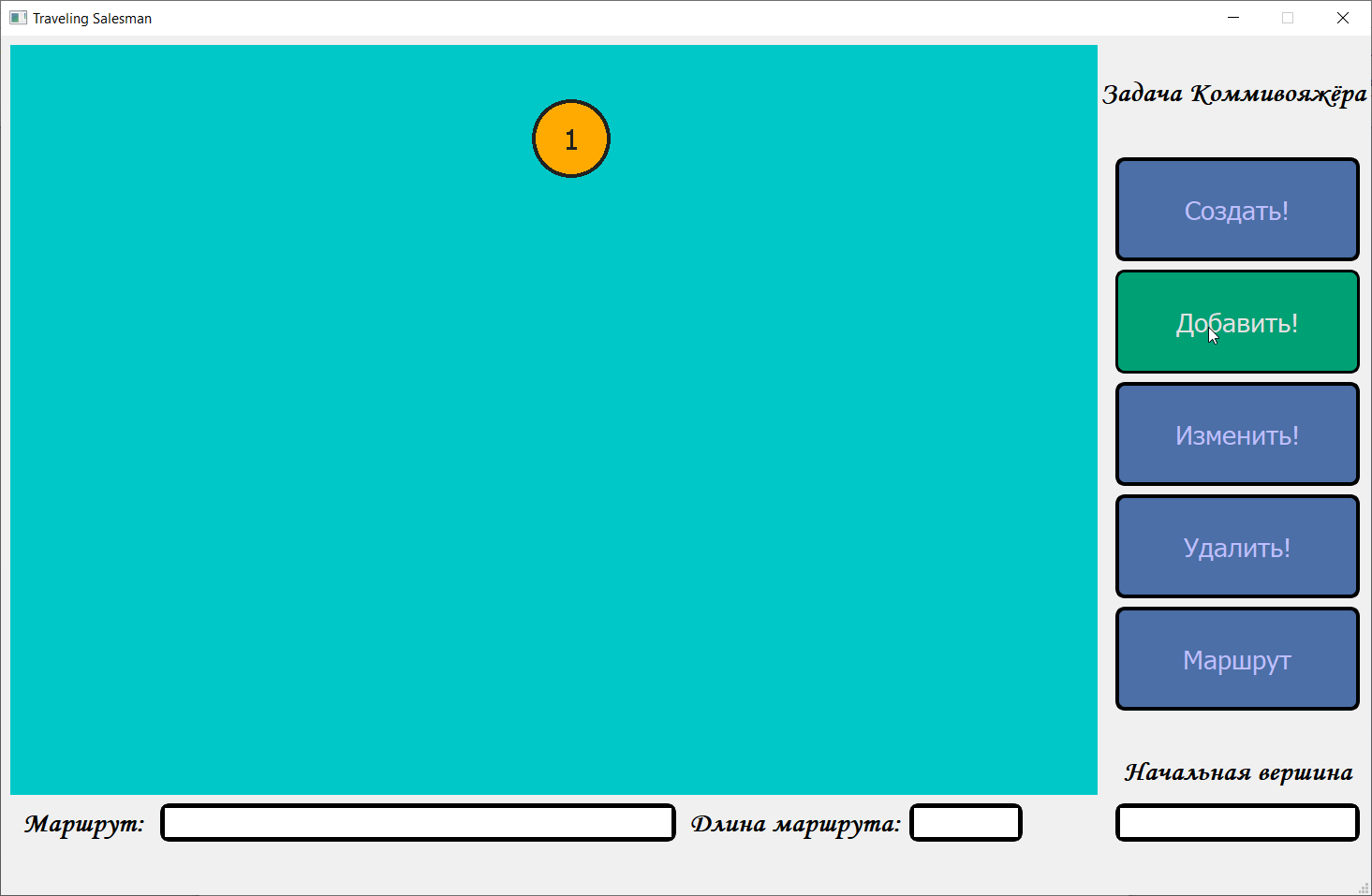
Рис. 2. UML-диаграмма задачи Коммивояжёра

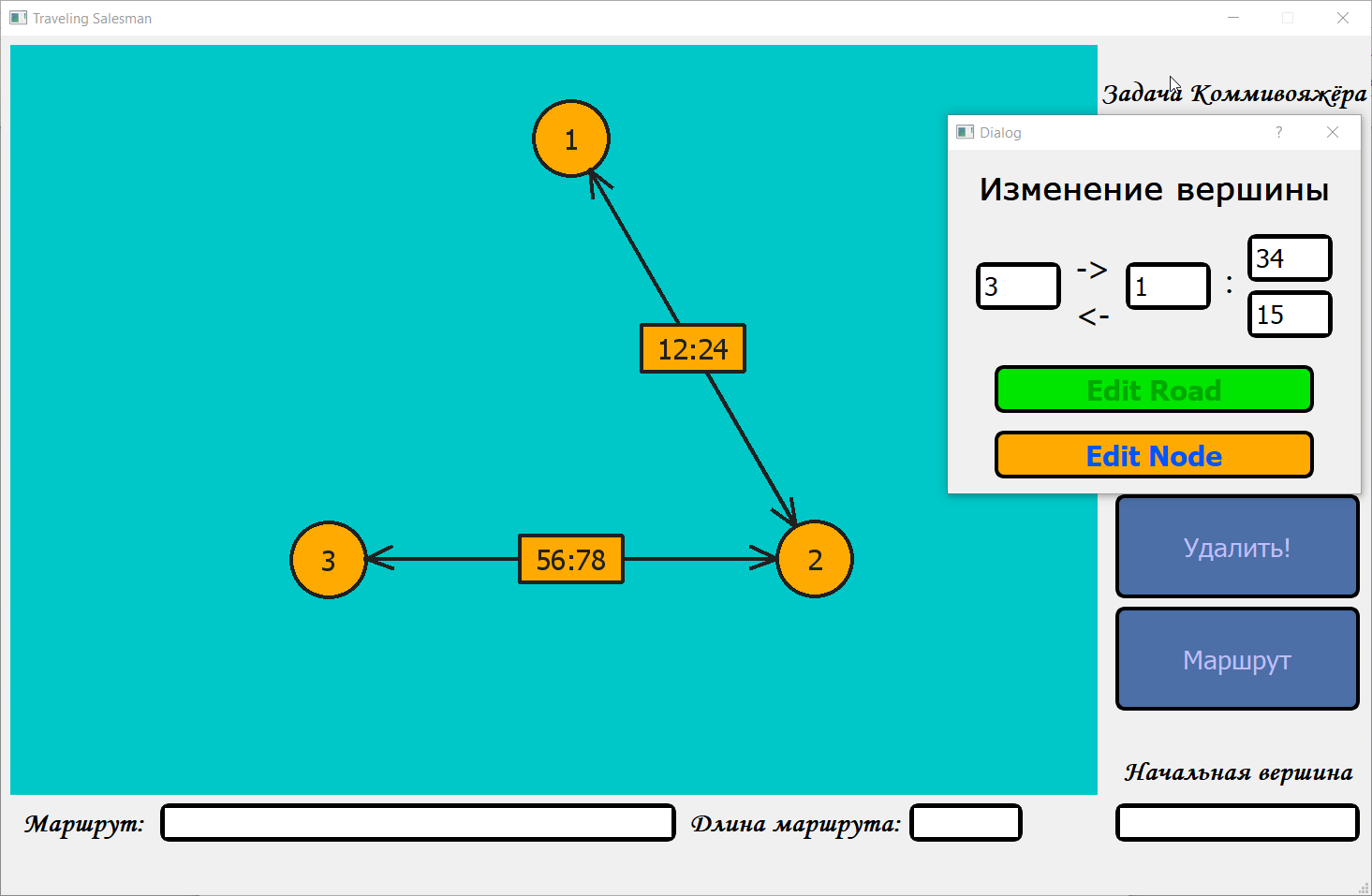
* 1. Класс QtWindow – главное окно программы. Отвечает за реализацию нажатий кнопок, полей ввода / вывода и визуализации графа для взаимодействия программы с пользователем. Имеет связь ассоциация с классами MyGLWidget, AddWindow, EditWindow, DeleteWindow и TravellerSalesMan.
  2. Класс MyGLWidget – переопределённый OpenGL-виджет. Отвечает за создания поля для помещения визуализации графа. Имеет связь ассоциация с классом QtWindow и MyPainter.
  3. Класс MyPainter – поле для рисования. Отвечает непосредственно за визуализацию графа. Имеет связь ассоциация с классом MyGLWidget.
  4. Класс AddWindow – диалоговое окно. Отвечает за добавление нового узла графа. Имеет связь ассоциация с классом QtWindow.
  5. Класс EditWindow – диалоговое окно. Отвечает за изменение дорог узлов графа. Имеет связь ассоциация с классом QtWindow.
  6. Класс DeleteWindow – диалоговое окно. Отвечает за удаление узла графа. Имеет связь ассоциация с классом QtWindow.
  7. Класс TravellerSalesMan – алгоритм решения задачи Коммивояжёра. Отвечает за поиск оптимального маршрута и его длины. Имеет связь ассоциация с классом QtWindow.

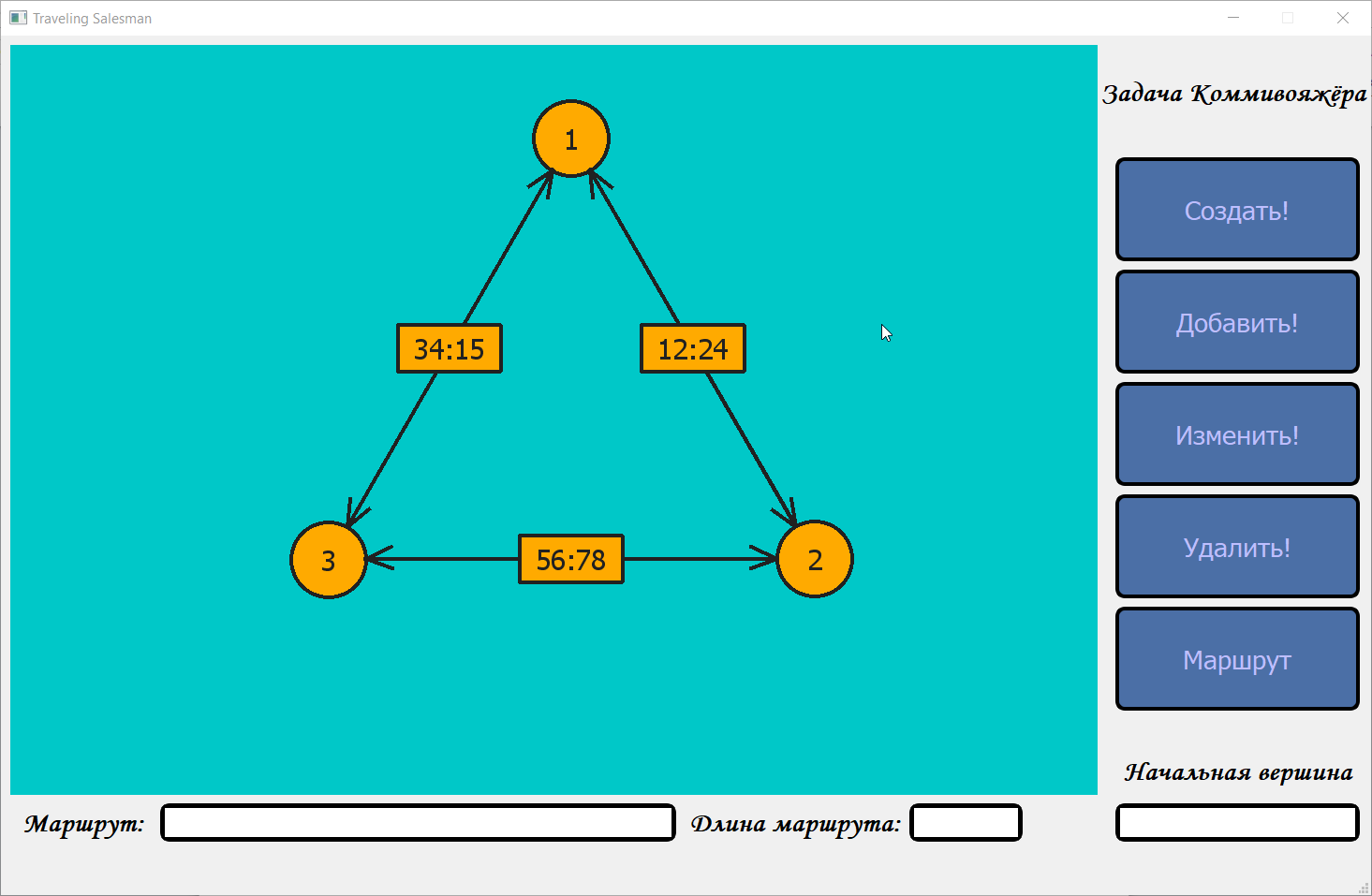
**Тестирование работы программы**

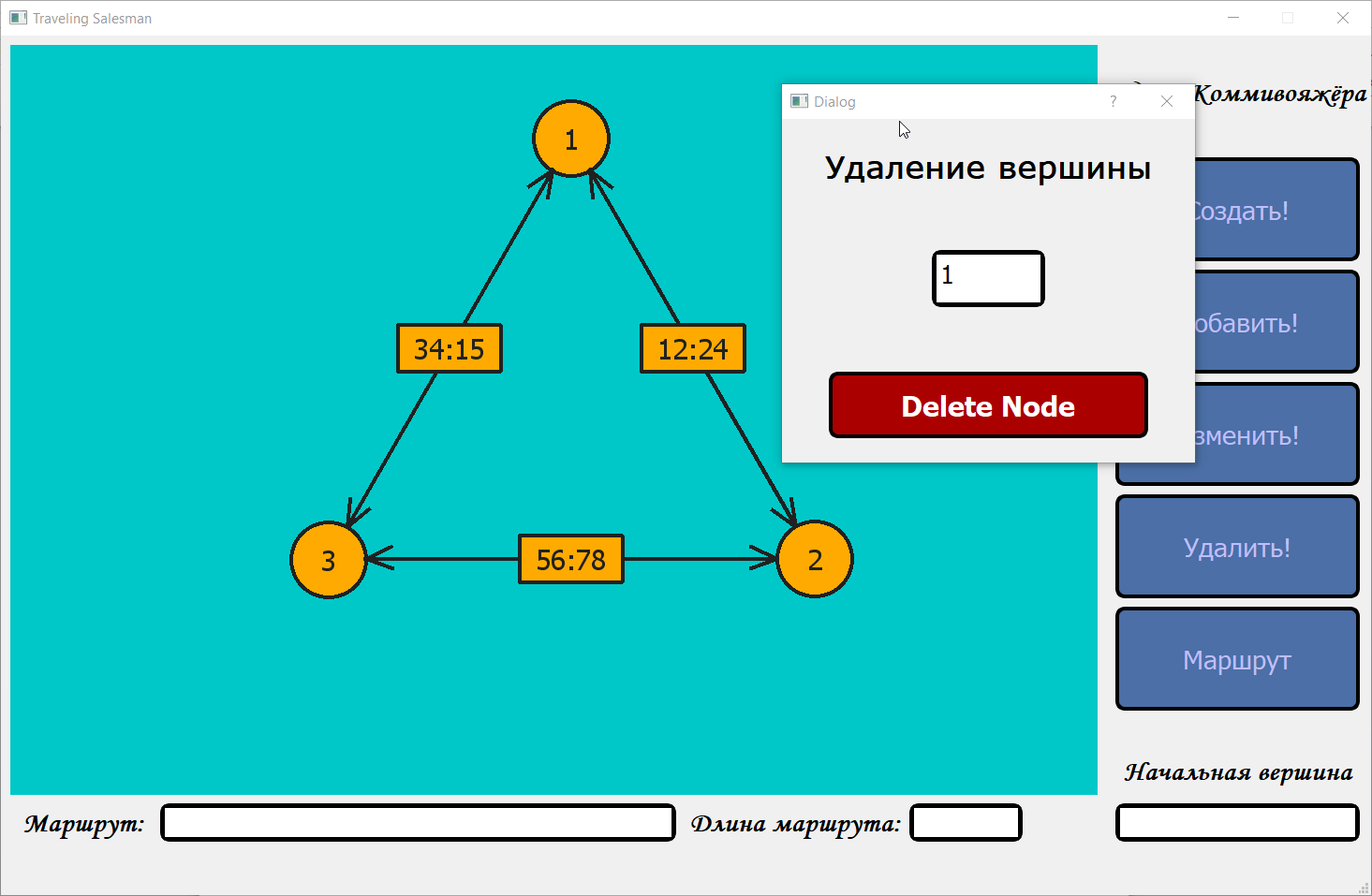
Ниже приведены скриншоты работы программы:

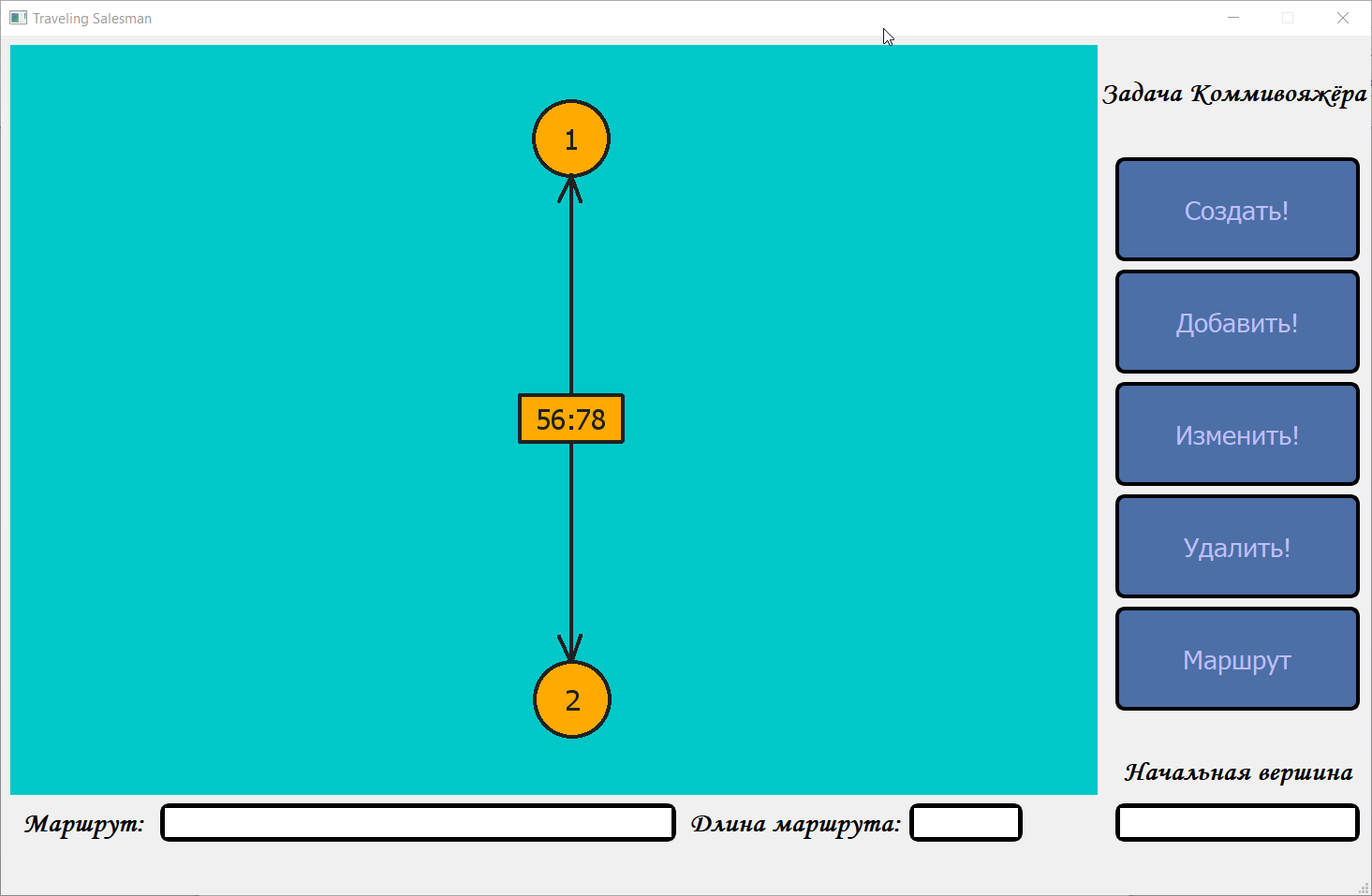


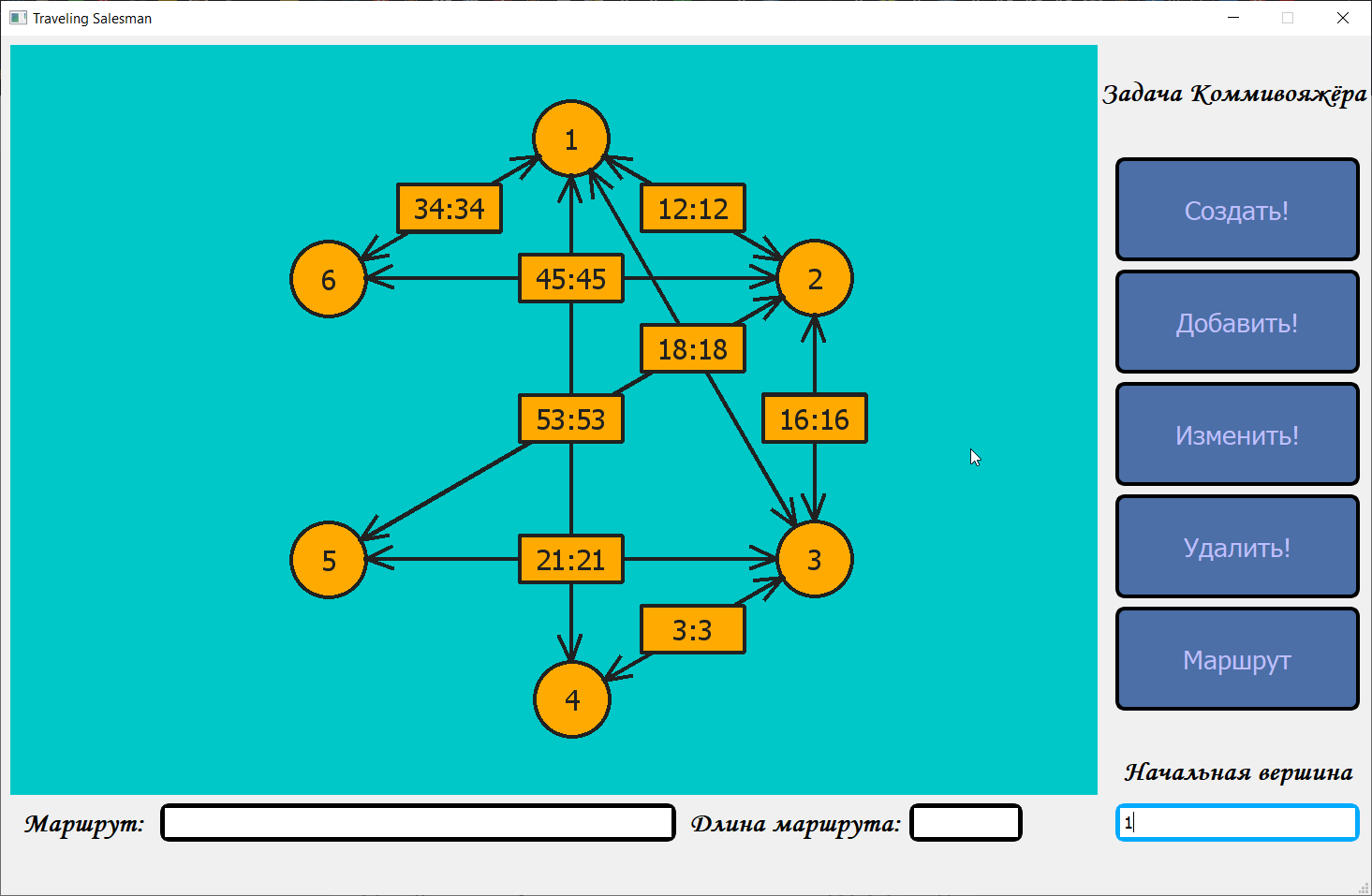


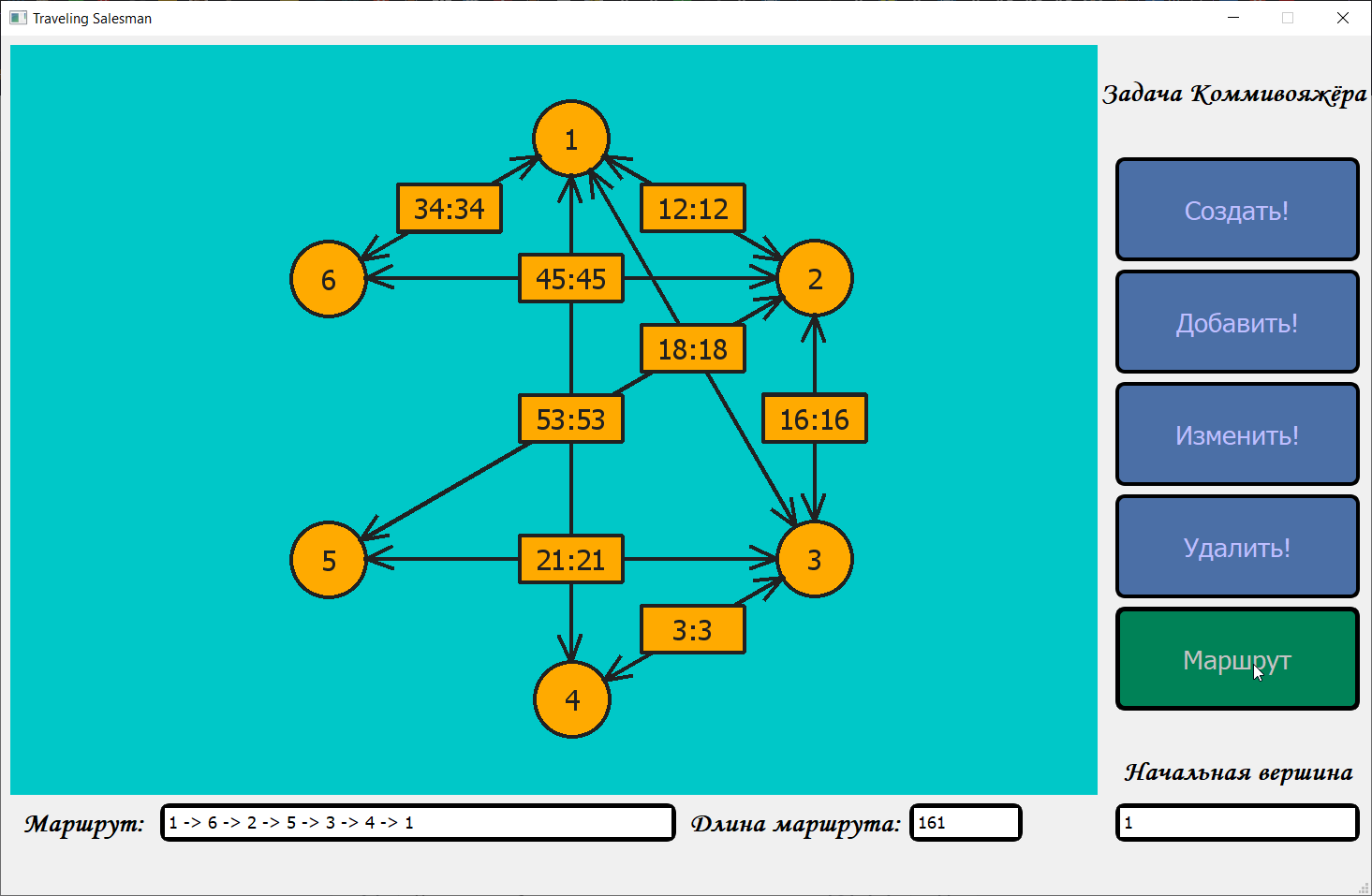


****



****

****

****

**Используемые средства**

Qt Creator 4.14.2 (реализация задач)

Visual Studio 2019(реализация дополнительных файлов)

OBS Studio 26.1.1 (запись видеоролика)