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»

Творческая работа № 1/2
по дисциплине
«Основы алгоритмизации и программирования»
на тему
«Задача Коммивояжера» и «Разработка АРМ специалиста»
Вариант №6

Выполнил студент гр. ИВТ-23-1б
Бучинский Ян Викторович

Проверил:
доцент кафедры ИТАС
Яруллин Денис Владимирович

_____ _____
(оценка) (подпись)

_____
(дата)

г. Пермь, 2024

Постановка задачи

Коммивояжер:

1. В качестве варианта для демонстрации работы программы взять свой вариант задания из лабораторной работы «ГРАФЫ» (не менее 6 вершин, двунаправленный граф). Модифицировать граф таким образом, чтобы для этого графа можно было решить задачу Коммивояжера. Можно придумать собственную альтернативную задачу, которую можно решить методом ветвей и границ. Это может быть игра, построенная по типу пошаговых настольных игр, к примеру. Разработать программу, которая будет универсальной на любом наборе исходных данных.
2. Проработать визуализирующую часть в программе средствами OpenGL или иных открытых кроссплатформенных графических библиотек в части построения графа. Интересные дизайнерские и конструкторские решения в интерфейсе применить: добавление новых узлов, перемещение узлов, установка связей между узлами, разрыв связей и прочие варианты демонстрации своего таланта.
3. Исходные данные должны приниматься с консоли, либо через графический интерфейс с помощью Qt, Windows Forms или других фреймворков и библиотек в экосистеме языка C++.
4. Задокументировать программу диаграммой классов UML.

АРМ специалиста:

1. Разработка приложения способного помочь специалисту в его работе.

Анализ Задач

Коммивояжер:

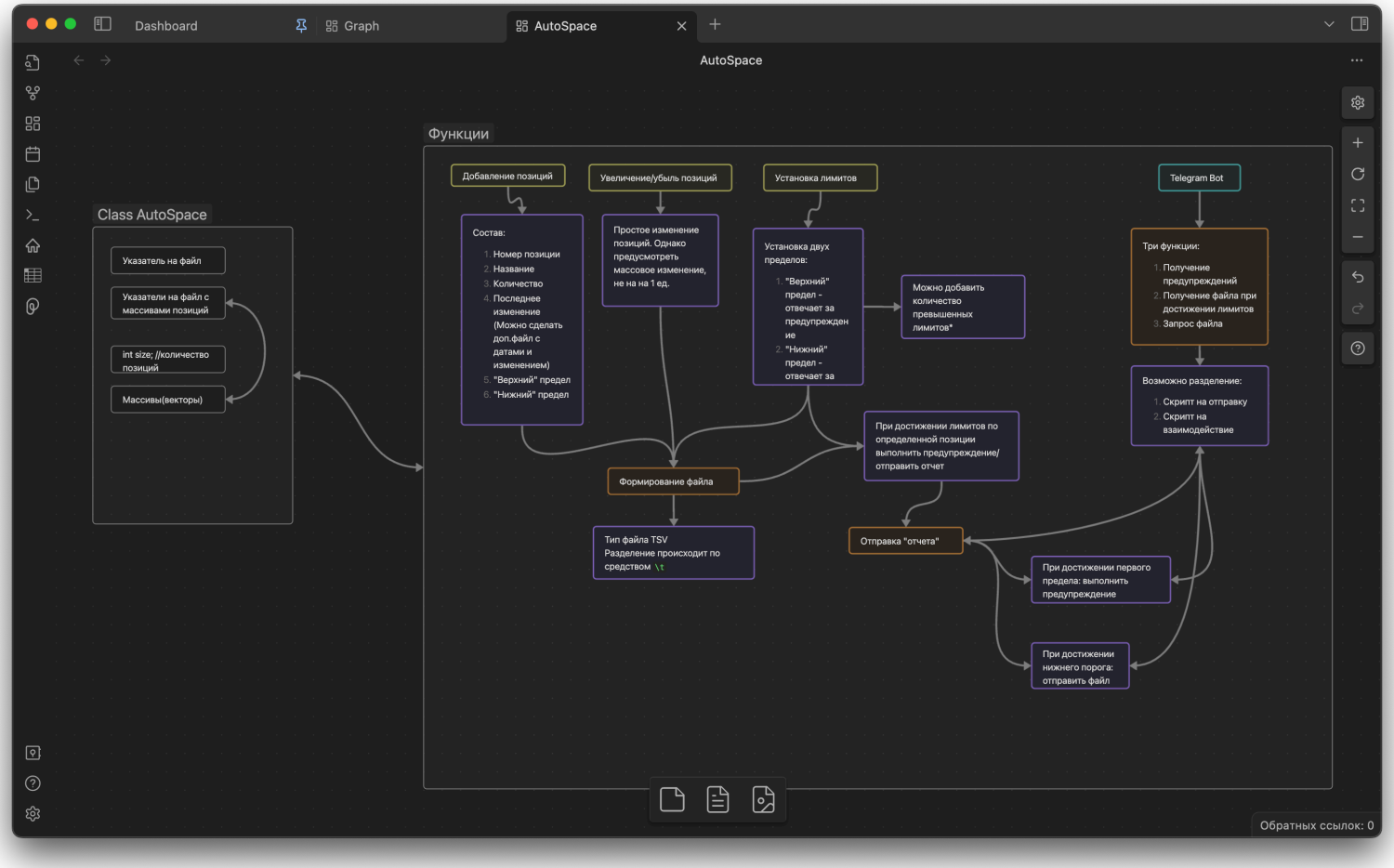
1. Инициализация:
   - Создается граф, представленный матрицей смежности.
   - Для каждой вершины графа создается объект типа Node, который содержит информацию о предыдущих вершинах, текущей вершине, суммарном весе пути до текущей вершины и указателе на граф.
2. Рекурсивный процесс:
   - На каждом шаге выбирается вершина с наименьшей верхней оценкой, которая еще не была посещена.
   - Для выбранной вершины генерируются все возможные следующие вершины, и для каждой из них вычисляются верхняя и нижняя оценки.
   - Выбирается следующая вершина с наименьшей верхней оценкой и добавляется в список рассматриваемых вершин.
   - Процесс продолжается до тех пор, пока не будет рассмотрено все пространство возможных маршрутов.
3. Поиск решения:
   - На каждом шаге алгоритма обновляется текущий рекорд - наименьшая из верхних оценок.
   - Если нижняя оценка текущей вершины больше текущего рекорда, ветвь обрывается.
   - Если нижняя оценка равна верхней, исследование этой ветви завершается.
   - После обхода всех ветвей в поиске решения возвращается наименьший найденный рекорд.
4. Вывод результата:
   - В конце алгоритма выводится оптимальный маршрут - последовательность посещенных вершин с их весами и матрица переходов между ними.

АРМ специалиста: **Наработки**

1. Тема:Разработка CPM системы для контроля позиций на складах.
   a. Добавление позиций,
   b. Увеличение/Уменьшение количества определенной позиции,
   c. Установка двух уровней лимитов
      1) «Верхний»: Отсылка предупреждения в бота,
      2) «Нижний»: Отсылка файла с текущими позициями и оповещение о достижение критического значения по определенной позиции.
   d. Ручной запрос файла TSV через бота.
2. Работа программы разбита на 3 основных пласта:
   1) Работа с файлами и обработка информации «под капотом»,

2) 3 скрипта для работы с ботом Telegram,
3) Визуализация программы.

3. Работа с файлами будет в свою очередь разбита еще на два этапа:
   1) Тк вся обработка информации будет завязана на работе с векторами их будет необходимо бинарно записывать в специально отведенные файлы (с помощью stdio.h). Чтение будет также происходить бинарно. Данные действия позволят снизить нагрузку на память,
   2) Второй же этап заключается в формировании файла TSV, тк вся логика программы будет завязана на векторах и их бинарном чтении и записи, файл TSV будет использован в свою очередь для удобного отображения позиций для пользователя (TSV файл можно будет легко открывать через Excel или подобные программы). В свою очередь именно файл TSV будет использоваться для отправки пользователю через бота Telegram.
4. Скрипты для бота:
   1) Отправка предупреждений,
   2) Отправка файла TSV,
   3) Обработка запросов от пользователя.
5. Визуализация (на QT) будет представлять из себя список текущих позиций с возможность изменять их количество, добавлять новые и тд.



UML - диаграммы

Коммивояжер: Рисунок 1

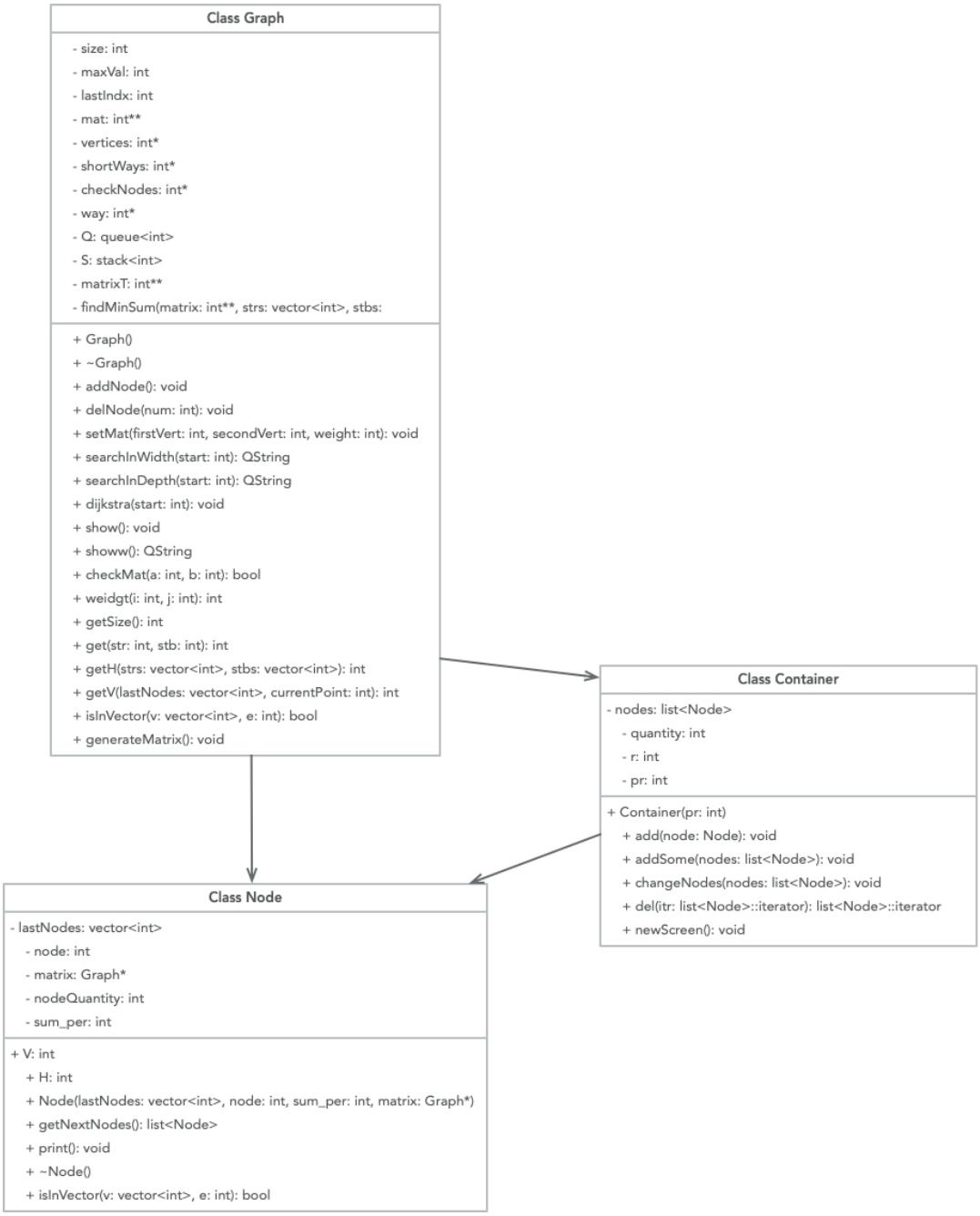
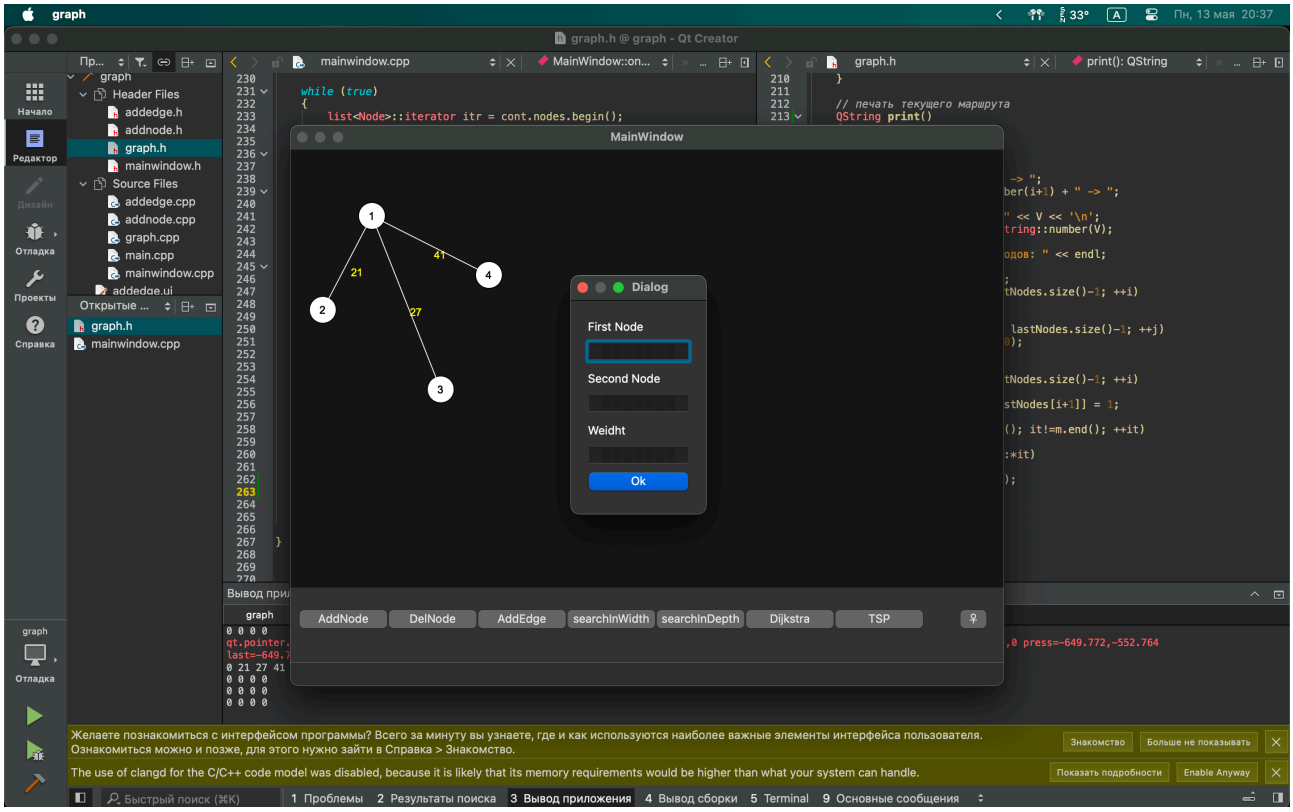


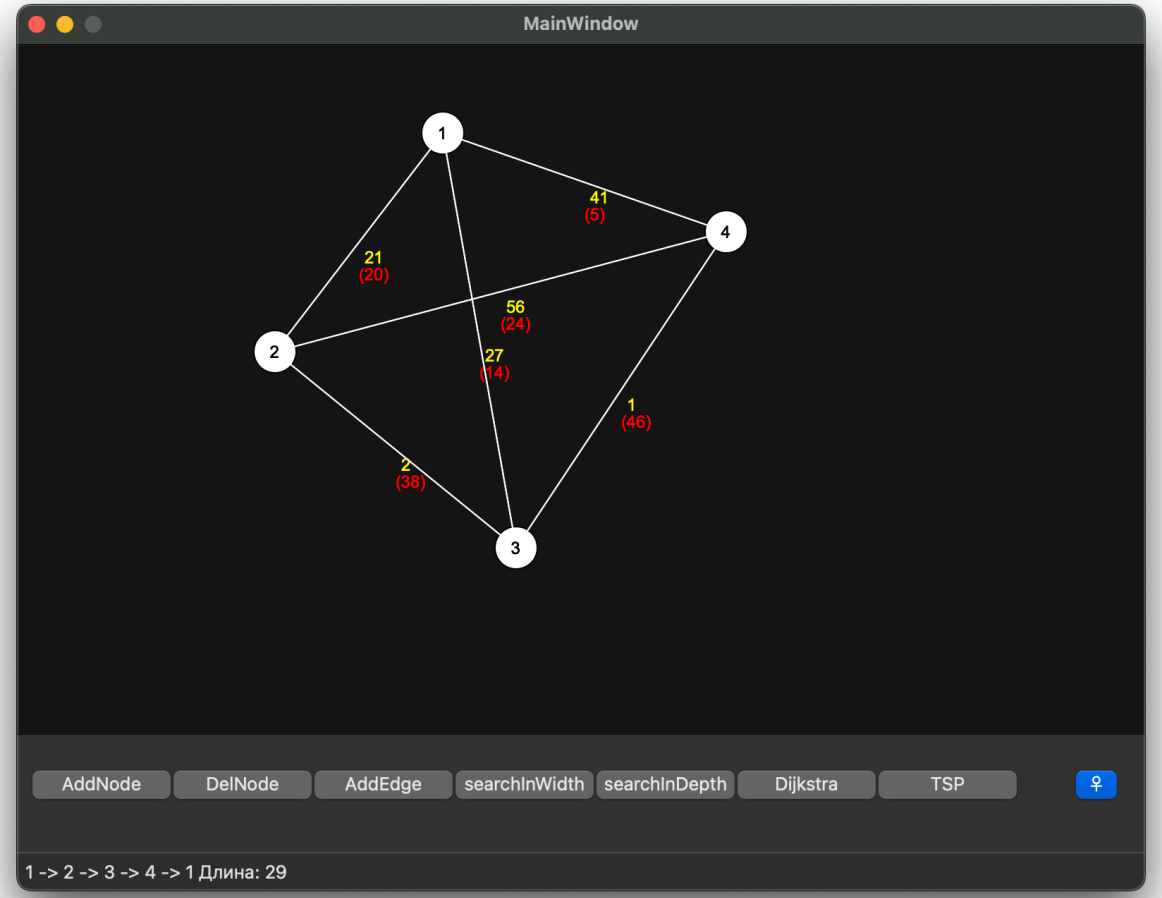
Рисунок 1

АРМ специалиста: **В разработке**

Демонстрация работы

Коммивояжер:





АРМ специалиста: **В разработке**

GitHub

https://github.com/MouseViolin/Labs_PSTU_2023/tree/main/Sem 2/Labs