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»

Лабораторная работа
по дисциплине «Основы алгоритмизации и программирования» на тему «Разработка автоматизированного рабочего места кухни ресторана. Задача коммивояжера»

Выполнила студентка группы ИВТ-23-1б
Долганова Диана Евгеньевна
Проверил:
доцент кафедры ИТАС
Яруллин Денис Владимирович

_____ _____
(оценка) (дата)

_____
(подпись)

г. Пермь, 2024

Автоматизированное рабочее место кухни ресторана

Постановка задачи

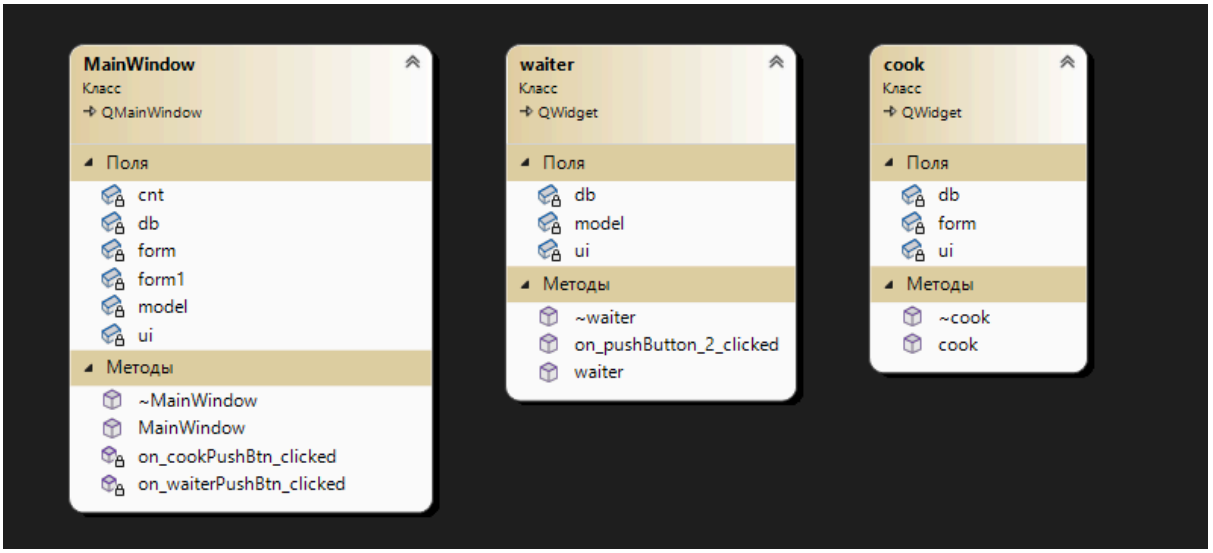
Разработать АРМ кухни ресторана, используя QT. Реализовать:

1. Добавление меню
2. Окно официанта, внутри которого пользователь выбирает заказанные блюда и их количество, после чего отправляет их повару.
3. Окно повара, которое принимает информацию из окна официанта. Внутри окна пользователь отмечает выполненные заказы, после чего окно очищается, и статус заказа передается в окно официанта.

Анализ задач

1) Класс mainwindow. Добавление окна с меню, ввод меню пользователем. Введенное меню записывается в базу данных и выводится на экран. Открытие окна официант и/или повара по нажатию на соответствующую кнопку.
2) Класс waiter. Добавление окна официанта.Значение из базы данных заносятся в список, из которого пользователь выбирает заказанные блюда и их количество. Полученная информация записывается в базу данных и отправляется в окно повара.
3) Класс cook. Добавление окна повара. Принимаются значения, полученные из окна официанта. Пользователь отмечает степень готовности заказа и отправляет информацию в окно официанта.

UML - диаграмма



Задача коммивояжера

Постановка задачи

Реализовать решение задачи коммивояжера, используя QT. Задача коммивояжера - комбинаторная задача оптимизации, в которой требуется найти самый короткий путь, проходящий через все заданные города (вершины) ровно один раз и возвращающийся в исходный город.

Анализ задач

1) Класс Graph

   1. Метод `addNode()` используется для добавления нового узла в граф. Он динамически корректирует размер матрицы смежности и связанных массивов для учета нового узла.

   2. Метод `delNode(int num)` используется для удаления узла из графа. Он обновляет матрицу смежности и связанные массивы для удаления указанного узла.

   3. Метод `show()` выводит матрицу смежности графа в консоль.

   4. Метод `setMat(int firstVert, int secondVert, int weight)` задает вес ребра между двумя вершинами в графе.

   5. Метод `searchInWidth(int start)` выполняет поиск в ширину, начиная с указанной вершины, и возвращает посещенные вершины в порядке обхода.

   6. Метод `searchInDepth(int start)` выполняет поиск в глубину, начиная с указанной вершины, и возвращает посещенные вершины в порядке обхода.

   7. Метод `dijkstra(int start)` реализует алгоритм Дейкстры для нахождения кратчайших путей, начиная с указанной вершины. Он вычисляет кратчайшие расстояния и путь ко всем остальным узлам.

8. Метод `showw()` возвращает кратчайший путь, вычисленный алгоритмом Дейкстры, в формате, который может быть использован для дальнейшей обработки.

2) Класс addedge

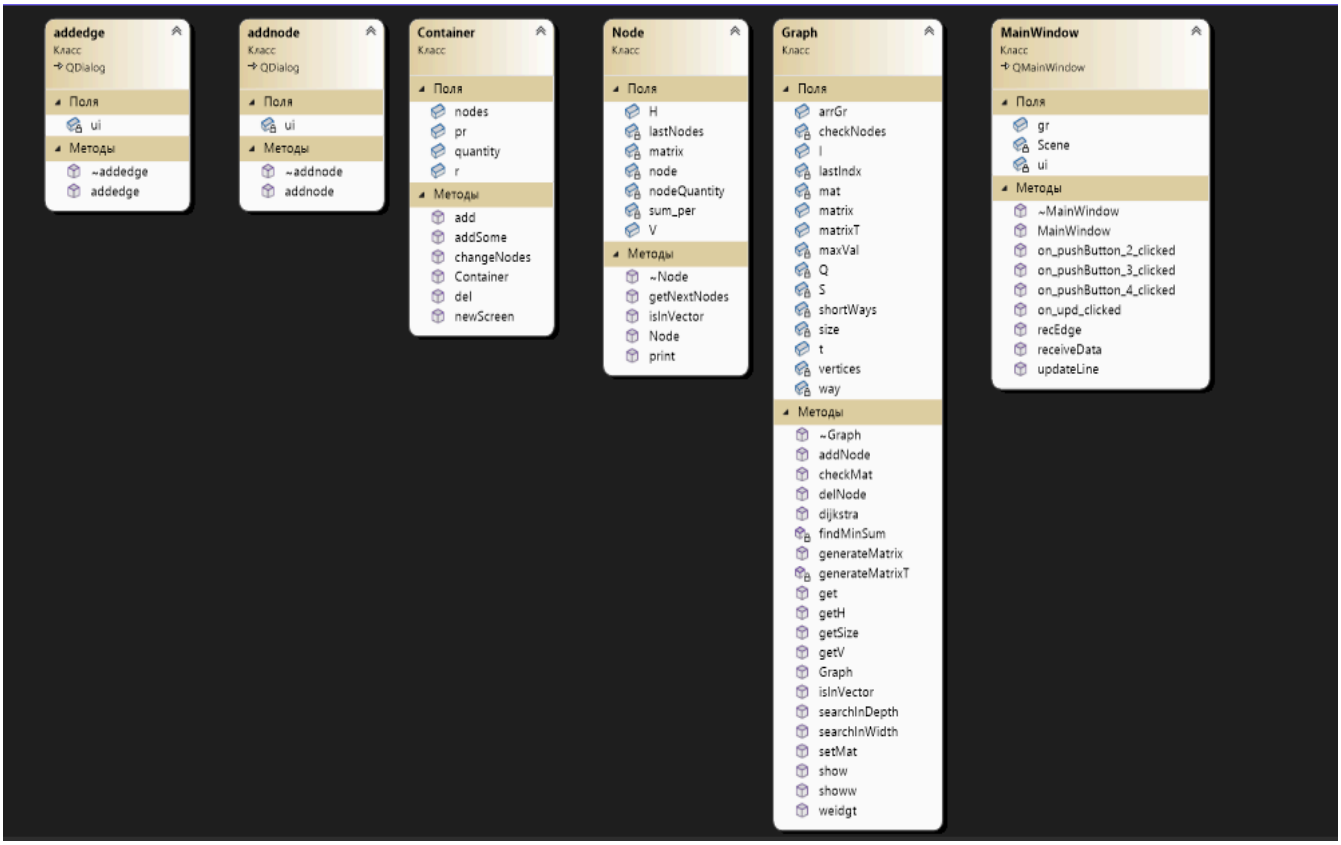
Метод on_pushButton_clicked() вызывается при нажатии на кнопку pushButton в пользовательском интерфейсе. В этом методе данные из трех текстовых полей (lineEdit_3, lineEdit_2, lineEdit) извлекаются в переменные типа QString, представляющие первый узел, второй узел и вес связи соответственно.

3) Класс addnode

Метод on_pushButton_clicked() запускается при нажатии на кнопку pushButton. В этом методе происходит:

1. Извлечение текстовых данных из трех полей ввода (lineEdit_3, lineEdit_2, lineEdit), представляющих первый узел, второй узел и вес связи, в объекты QString.
2. Отправка сигнала dataR(first, second, weight) с данными первого узла, второго узла и веса связи другой части программы для обработки.
3. Закрытие текущего диалогового окна с помощью метода close() после отправки данных.

UML - диаграмма



Демонстрация работы

https://youtu.be/MjOjaZIIz54?si=OX7PEdniVkVvwsPj

Ссылка на GitHub

https://github.com/dolganovadd/LABS_PSTU/tree/main/sem_2.gitkeep/labs.gitkeep/creative%20works.gitkeep