Министерство науки и высшего образования Российской Федерации

Федеральное государственное автономное образовательное учреждение высшего образования

«Пермский национальный исследовательский политехнический университет» (ПНИПУ)

Электротехнический факультет

Кафедра «Информационные технологии и автоматизированные системы» (ИТАС)

Лабораторная работа

по темам

«Задача коммивояжера»

«Автоматизация рабочего места»

Вариант 2

Выполнил

Студент группы ИВТ-23-1б

Адаев Даниил Дмитриевич

Проверил

Доцент кафедры ИТАС

Полякова О. А.

г. Пермь, 2024

Требования к Задаче коммивояжера:

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

Анализ задачи:

- Построение матрицы с исходными данными — в таблицу заносятся расстояния между городами (в ячейки типа A-A ставится -1 — значение не учитывается); при этом строкам соответствуют города отбытия, а столбцам города прибытия
- Нахождение минимумов по строкам — в каждой строке определяется минимальное число и выписывается в отдельный столбец
- Редукция строк – из значений ячеек каждой строки вычитаем соответствующий минимум, не затрагивая при этом клетки с бесконечным значением
- Редукция столбцов – то же самое, что и редукция строк, только применительно к столбцам
- Вычисление оценок нулевых клеток — считаем оценки для каждой ячейки с нулями, как сумму минимумов по строке и столбцу, в которых

располагается нулевая клетка, не учитывая при этом саму нулевую клетку

- Выбор нулевой клетки с максимальной оценкой — ищем среди нулевых клеток обладающую наибольшей оценкой (если таких ячеек несколько, выбираем любую), и получаем пару ветвей (вариантов) решения задачи: с включением в маршрут отрезка пути, относящегося к выбранной ячейке и без включения
- Редукция матрицы — вычеркиваем относящиеся к выбранной клетке строку и столбец, а также заменяем значение ячейки, соответствующей обратному пути.
- Записываем выбранную клетку и повторяем итерации с нахождения минимумов, пока не закончатся значащие ячейки в таблице.
- Получаем значения ребер графов: узлы, с которыми соединено ребро, которые остается лишь подставить друг за другом.

Код на c++

TS.pro

```
QT       += core gui

greaterThan(QT_MAJOR_VERSION, 4): QT += widgets

CONFIG += c++11

# You can make your code fail to compile if it uses deprecated APIs.
# In order to do so, uncomment the following line.
#DEFINES += QT_DISABLE_DEPRECATED_BEFORE=0x060000    # disables all the APIs
deprecated before Qt 6.0.0

SOURCES += \
    main.cpp \
    tsp.cpp

HEADERS += \
    tsp.h

FORMS += \
    tsp.ui

# Default rules for deployment.
qnx: target.path = /tmp/${TARGET}/bin
else: unix:!android: target.path = /opt/${TARGET}/bin
       !isEmpty(target.path): INSTALLS += target
```

tsp.h

```
#pragma once
#include <QMainWindow>
#include "ui_tsp.h"
#include <QWidget>
```

```
#include <QPushButton>
#include <QMouseEvent>
#include <unordered_map>
#include <unordered_set>

using namespace std;

class Edge;

struct Node
{
    int data;
    vector<Edge*> edges;
    QPoint pos;
    Node() { pos = QPoint(350, 250); }
};

struct Edge
{
    int weight;
    Node* to;
    Node* from;
};

struct Graph
{
    vector<Node*> vnodes;

    unordered_map<int, Node*> nodes;

    void addNode(int data);
    void addEdge(int fromData, int toData, int weight);
    void clearGraph();
    void updateEdgeWeight(int startData, int endData, int newWeight);

    void removeNode(int data);
    void removeEdge(int startData, int endData);

    int** createNodeMap ();
    void min_line(int** map);
    void min_column(int** map);
    int min_str(int** map, int i, int l);
    int min_stl(int** map, int i, int l);
    Edge* clear_map(int** map);
    vector<int> TSPsolve();
    int way(vector<int> path);
};

class TSP : public QMainWindow
{
    Q_OBJECT

public:
    TSP(QWidget *parent = nullptr);
    ~TSP();
    Graph graph;

protected:
    void paintEvent(QPaintEvent* event) override;
    void mousePressEvent(QMouseEvent* event) override;
    void mouseMoveEvent(QMouseEvent* event) override;
    void mouseReleaseEvent(QMouseEvent* event) override;

private:
    Ui::TSP *ui;
    Node* m_selectedNode;
```

```
    bool m_nodeSelected;

    bool sel = 0;
    Node* sNode;

    void on_pushButton_clicked();
    void on_pushButton_2_clicked();
    void on_pushButton_3_clicked();
    void on_pushButton_4_clicked();
    void on_pushButton_5_clicked();

    void on_pushButton_6_clicked();
    void on_pushButton_7_clicked();
    void on_pushButton_8_clicked();
      };
```

main.cpp

```
#include "tsp.h"

#include <QApplication>

int main(int argc, char *argv[])
{
    QApplication a(argc, argv);
    TSP w;
    w.show();
    return a.exec();
}
```

tsp.cpp

```
#include "tsp.h"
#include "ui_tsp.h"
#include <cmath>
#include <QPainter>

void Graph::addNode(int data)
{
    if (nodes.find(data) == nodes.end())
    {
        Node* newNode = new Node;
        newNode->data = data;
        nodes[data] = newNode;
        vnodes.push_back(newNode);//
    }
}


void Graph::addEdge(int fromData, int toData, int weight)
{
    for (Edge* edge : nodes[fromData]->edges)
    {
        if (edge->to == nodes[toData])
        {
            return;
        }
    }

    Edge* newEdge = new Edge();
    newEdge->to = nodes[toData];
    newEdge->from = nodes[fromData];
    newEdge->weight = weight;
```

```
    nodes[fromData]->edges.push_back(newEdge);
}

void Graph::clearGraph()
{
    for (auto& pair : nodes)
    {
        Node* node = pair.second;
        delete node;
    }
    nodes.clear();
    vnodes.clear();
}

void Graph::updateEdgeWeight(int startData, int endData, int newWeight)
{
    if (nodes.find(startData) == nodes.end() || nodes.find(endData) ==
nodes.end())
    {
        return;
    }

    Node* startNode = nodes[startData];
    Node* endNode = nodes[endData];

    for (Edge* edge : startNode->edges)
    {
        if (edge->to == endNode)
        {
            edge->weight = newWeight;
            return;
        }
    }
}

void Graph::removeNode(int data)
{
    for (auto& pair : nodes)
    {
        Node* node = pair.second;
        vector<Edge*> edges_to_remove;

        for (Edge* edge : node->edges)
        {
            if (edge->to->data == data)
            {
                edges_to_remove.push_back(edge);
            }
        }

        for (Edge* edge : edges_to_remove)
        {
            auto it = find(node->edges.begin(), node->edges.end(), edge);
            if (it != node->edges.end())
            {
                node->edges.erase(it);
                delete edge;
            }
        }
    }

    auto it = nodes.find(data);
    if (it != nodes.end())
    {
        delete it->second;
        nodes.erase(it);
```

```
    }

    Node* nodeToRemove = nullptr;//
    for (Node* nodeTo : vnodes)
    {
        if (nodeTo->data == data)
        {
            nodeToRemove = nodeTo;
            break;
        }
    }

    if (nodeToRemove)
    {

        auto itn = find(vnodes.begin(), vnodes.end(), nodeToRemove);
        if (itn != vnodes.end())
        {
            vnodes.erase(itn);
        }
    }
}

void Graph::removeEdge(int startData, int endData)
{
    auto startNodeIt = nodes.find(startData);
    auto endNodeIt = nodes.find(endData);

    if (startNodeIt == nodes.end() || endNodeIt == nodes.end())
    {
        return;
    }

    Node* startNode = startNodeIt->second;
    Node* endNode = endNodeIt->second;

    Edge* edgeToRemove = nullptr;

    for (Edge* edge : startNode->edges)
    {
        if (edge->to->data == endData)
        {
            edgeToRemove = edge;
            break;
        }
    }

    if (edgeToRemove)
    {
        auto it = find(startNode->edges.begin(), startNode->edges.end(),
edgeToRemove);
        if (it != startNode->edges.end())
        {
            startNode->edges.erase(it);
            delete edgeToRemove;
        }
    }
}

TSP::TSP(QWidget *parent): QMainWindow(parent), ui(new Ui::TSP)
{
    ui->setupUi(this);

    connect(ui->pushButton, &QPushButton::clicked, this,
&TSP::on_pushButton_clicked);
```

```
    connect(ui->pushButton_2, &QPushButton::clicked, this,
&TSP::on_pushButton_2_clicked);
    connect(ui->pushButton_3, &QPushButton::clicked, this,
&TSP::on_pushButton_3_clicked);
    connect(ui->pushButton_4, &QPushButton::clicked, this,
&TSP::on_pushButton_4_clicked);
    connect(ui->pushButton_5, &QPushButton::clicked, this,
&TSP::on_pushButton_5_clicked);

    connect(ui->pushButton_6, &QPushButton::clicked, this,
&TSP::on_pushButton_6_clicked);
    connect(ui->pushButton_7, &QPushButton::clicked, this,
&TSP::on_pushButton_7_clicked);
    connect(ui->pushButton_8, &QPushButton::clicked, this,
&TSP::on_pushButton_8_clicked);
}

TSP::~TSP()
{
    delete ui;
}

void TSP::paintEvent(QPaintEvent* event)
{
    QPainter painter(this);
    QFont font = painter.font();
    font.setPointSize(16);
    painter.setFont(font);
    for (const auto& pair : graph.nodes)
    {
        Node* node = pair.second;
        for (Edge* edge : node->edges)
        {
            QPoint pos_f;
            QPoint pos_t;
            int d = 20 * sin(atan(1));
            double angles = atan2(-(edge->to->pos.y() - node->pos.y()),
(edge->to->pos.x() - node->pos.x()));
            pos_f = QPoint(node->pos.x() + 20 * cos(angles), node->pos.y() -
20 * sin(angles));
            pos_t = QPoint(edge->to->pos.x() - 20 * cos(angles), edge->to-
>pos.y() + 20 * sin(angles));
            painter.drawLine(pos_f, pos_t);
            int x_t = pos_f.x() + 4 * (pos_t.x() - pos_f.x()) / 5;
            int y_t = pos_f.y() - 4 * (pos_f.y() - pos_t.y()) / 5;
            painter.drawText(x_t - 10, y_t + 10, QString::number(edge-
>weight));

            QLine line(pos_f, pos_t);
            double angle = atan2(-line.dy(), line.dx()) - M_PI / 2;
            double arrowSize = 15;
            double arrowLength = 20;
            QPointF arrowP1 = pos_t + QPointF(sin(angle - M_PI / 10) *
arrowSize, cos(angle - M_PI / 10) * arrowSize);
            QPointF arrowP2 = pos_t + QPointF(sin(angle + M_PI / 10) *
arrowSize, cos(angle + M_PI / 10) * arrowSize);
            QPolygonF arrowHead;
            arrowHead << pos_t << arrowP1 << arrowP2;
            painter.drawPolygon(arrowHead);
        }
    }

    for (const auto& pair : graph.nodes)
    {
        Node* node = pair.second;
        painter.drawEllipse(node->pos, 20, 20);
```

```
        painter.drawText(node->pos.x() - 9, node->pos.y() + 8,
QString::number(node->data));
    }
    if (sel)
    {
        painter.drawEllipse(100, 100, 40, 40);
        painter.setBrush(Qt::green);
        painter.drawEllipse(sNode->pos, 20, 20);
        painter.drawText(sNode->pos.x() - 9, sNode->pos.y() + 8,
QString::number(sNode->data));
    }
}

void TSP::mousePressEvent(QMouseEvent* event)
{
    if (event->button() == Qt::LeftButton)
    {
        m_nodeSelected = false;
        for (const auto& pair : graph.nodes)
        {
            Node* node = pair.second;
            if ((event->pos() - node->pos).manhattanLength() < 30)
            {
                m_selectedNode = node;
                m_nodeSelected = true;
                break;
            }
        }
        update();
    }
}

void TSP::mouseMoveEvent(QMouseEvent* event)
{
    if (m_nodeSelected && m_selectedNode)
    {
        m_selectedNode->pos = event->pos();
        update();
    }
}
void TSP::mouseReleaseEvent(QMouseEvent* event)
{
    if (event->button() == Qt::LeftButton && m_nodeSelected)
    {
        m_nodeSelected = false;
        m_selectedNode = nullptr;
        update();
    }
}

int** Graph::createNodeMap ()//
{
    int** map = new int*[vnodes.size()];
        for (int i = 0; i < vnodes.size(); ++i)
        {
            map[i] = new int[vnodes.size()];
        }
    for (int i = 0; i < vnodes.size(); i++)
    {
        for (int j = 0; j < vnodes.size(); j++)
        {
            if (i != j)
            {
                bool f = false;
                for (Edge* e : vnodes[i]->edges)
                {
```

```
                    if (e->to == vnodes[j])
                    {
                        f = true;
                        map[i][j] = e->weight;
                        break;
                    }
                }
                if (f == false)
                {
                    map[i][j] = -1;
                }
            }
            else
            {
                map[i][j] = -1;
            }
        }
    }
    return map;
}

void Graph::min_line(int** map)
{
    int min;
    for (int i = 0; i < vnodes.size(); i++)
    {
        min = -1;
        for (int j = 0; j < vnodes.size(); j++)
        {
            if ((min > map[i][j] || min == -1) && map[i][j] >= 0)
            {
                min = map[i][j];
            }
        }
        for (int j = 0; j < vnodes.size(); j++)
        {
            if (map[i][j] != -1)
            {
                map[i][j] -= min;
            }
        }
    }
}

void Graph::min_column(int** map)
{
    int min;
    for (int i = 0; i < vnodes.size(); i++)
    {
        min = -1;
        for (int j = 0; j < vnodes.size(); j++)
        {
            if ((min > map[j][i] || min == -1) && map[j][i] >= 0)
            {
                min = map[j][i];
            }
        }
        for (int j = 0; j < vnodes.size(); j++)
        {
            if (map[j][i] != -1)
            {
                map[j][i] -= min;
            }
        }
    }
}
```

```
int Graph::min_str(int** map, int i, int l)
{
    int min = -1;
    for (int j = 0; j < vnodes.size(); j++)
    {
        if (j != l)
        {
            if ((min > map[i][j] || min == -1) && map[i][j] >= 0)
            {
                min = map[i][j];
            }
        }
    }
    if (min == -1)
    {
        min = 0;
    }

    return min;
}

int Graph::min_stl(int** map, int i, int l)
{
    int min = -1;
    for (int j = 0; j < vnodes.size(); j++)
    {
        if (j != l)
        {
            if ((min > map[j][i] || min == -1) && map[j][i] >= 0)
            {
                min = map[j][i];
            }
        }
    }
    if (min == -1)
    {
        min = 0;
    }

    return min;
}

Edge* Graph::clear_map(int** map)
{
    Edge* a = new Edge;
    int max = -1, k, m;
    for (int i = 0; i < vnodes.size(); i++)
    {
        for (int j = 0; j < vnodes.size(); j++)
        {
            if (map[i][j] == 0)
            {
                if (max <= min_str(map, i, j) + min_stl(map, j, i))
                {
                    max = min_str(map, i, j) + min_stl(map, j, i);
                    k = i;
                    m = j;
                }
            }
        }
    }

    if (max == -1)
    {
        return nullptr;
```

```
    }

    for (int i = 0; i < vnodes.size(); i++)
    {
        for (int j = 0; j < vnodes.size(); j++)
        {
            if (i == k)

            {
                map[i][j] = -1;
            }
            if (j == m)
            {
                map[i][j] = -1;
            }
        }
    }

    map[m][k] = -1;

    a->from = vnodes[k];
    a->to = vnodes[m];

    return a;
}

vector<int> Graph::TSPsolve()
{
    vector<Edge*> rez;
    vector<int> bestPath;
    int** map =createNodeMap();
    if (vnodes.size() == 2)
    {
        bestPath.push_back(vnodes[0]->data);
        bestPath.push_back(vnodes[1]->data);
        return bestPath;
    }

    int i = 0;
    while (i < vnodes.size())
    {
        min_line(map);
        min_column(map);
        rez.push_back(clear_map(map));
        i++;
    }

    for (Edge* e : rez)
    {
        if (e == nullptr)
        {
            bestPath.push_back(-1);
            return bestPath;
        }
    }

    bestPath.push_back(rez[0]->from->data);
    bestPath.push_back(rez[0]->to->data);
    for (int i = 2; i < vnodes.size(); i++)
    {
        for (int j = 1; j < rez.size(); j++)
        {
            if (rez[j]->from->data == bestPath[i - 1])
            {
                bestPath.push_back(rez[j]->to->data);
                break;
```

```
            }
        }
    }

    for (int i = 0; i < vnodes.size(); i++)
    {
        delete[] map[i];
    }
    delete[] map;

    return bestPath;
}

int Graph::way(vector<int> path)
{
    int w = 0;
    for (int i = 0; i < path.size() - 1; i++)
    {
        for (Node* n : vnodes)
        {
            if (n->data == path[i])
            {
                for (Node* nod : vnodes)
                {
                    if (nod->data == path[i + 1])
                    {
                        for (Edge* e : n->edges)
                        {
                            if (e->to == nod)
                            {
                                w += e->weight;
                            }
                        }
                    }
                }
            }
        }
    }

    for (Node* n : vnodes)
    {
        if (n->data == path[path.size() - 1])
        {
            for (Node* nod : vnodes)
            {
                if (nod->data == path[0])
                {
                    for (Edge* e : n->edges)
                    {
                        if (e->to == nod)
                        {
                            w += e->weight;
                        }
                    }
                }
            }
        }
    }

    return w;
}

void TSP::on_pushButton_clicked()
{
    QString text = ui->lineEdit->text();
    if (text.isEmpty()) return;
```

```
    int nodeValue = text.toInt();
    graph.addNode(nodeValue);
    ui->lineEdit->clear();
    update();
}

void TSP::on_pushButton_2_clicked()
{
    if (ui->lineEdit_2->text().isEmpty() or ui->lineEdit_3->text().isEmpty()
or ui->lineEdit_4->text().isEmpty())
    {
        return;
    }
    int fromNode = ui->lineEdit_2->text().toInt();
    int toNode = ui->lineEdit_3->text().toInt();
    int weight = ui->lineEdit_4->text().toInt();
    if (graph.nodes.find(fromNode) != graph.nodes.end() &&
graph.nodes.find(toNode) != graph.nodes.end())
    {
        graph.addEdge(fromNode, toNode, weight);
        ui->lineEdit_2->clear();
        ui->lineEdit_4->clear();
        ui->lineEdit_3->clear();
        update();
    }
}
void TSP::on_pushButton_3_clicked()
{
    if (ui->lineEdit_5->text().isEmpty())
    {
        return;
    }
    int del = ui->lineEdit_5->text().toInt();
    graph.removeNode(del);
    ui->lineEdit_5->clear();
    update();
}
void TSP::on_pushButton_4_clicked()
{
    if (ui->lineEdit_6->text().isEmpty() or ui->lineEdit_7->text().isEmpty())
    {
        return;
    }
    int s = ui->lineEdit_6->text().toInt();
    int f = ui->lineEdit_7->text().toInt();
    graph.removeEdge(s, f);
    ui->lineEdit_7->clear();
    ui->lineEdit_6->clear();
    update();
}
void TSP::on_pushButton_5_clicked()
{
    graph.addNode(1);
    graph.addNode(2);
    graph.addNode(3);
    graph.addNode(4);
    graph.addNode(5);
    graph.addNode(6);
    graph.addEdge(1, 2, 8);
    graph.addEdge(1, 6, 11);
    graph.addEdge(2, 3, 12);
    graph.addEdge(2, 5, 10);
    graph.addEdge(3, 4, 16);
    graph.addEdge(4, 5, 5);
    graph.addEdge(4, 6, 9);
```

```
    graph.addEdge(5, 6, 6);

    graph.addEdge(2, 1, 8);
    graph.addEdge(6, 1, 11);
    graph.addEdge(3, 2, 12);
    graph.addEdge(5, 2, 10);
    graph.addEdge(4, 3, 16);
    graph.addEdge(5, 4, 5);
    graph.addEdge(6, 4, 9);
    graph.addEdge(6, 5, 6);
    update();
}

void TSP::on_pushButton_6_clicked()
{
    if (ui->lineEdit_8->text().isEmpty() or ui->lineEdit_9->text().isEmpty()
or ui->lineEdit_10->text().isEmpty())
    {
        return;
    }
    int s = ui->lineEdit_8->text().toInt();
    int t = ui->lineEdit_9->text().toInt();
    int w = ui->lineEdit_10->text().toInt();
    graph.updateEdgeWeight(s, t, w);
    ui->lineEdit_8->text().clear();
    ui->lineEdit_9->text().clear();
    ui->lineEdit_10->text().clear();
    update();
}

void TSP::on_pushButton_7_clicked()
{
    graph.clearGraph();
    update();
}

void TSP::on_pushButton_8_clicked()
{
    ui->label_9->clear();
    ui->label_10->clear();
    ui->label_11->clear();

    if (graph.vnodes.size() == 0)
    {
        ui->label_9->setText("Невозможно решить задачу коммивояжера");
        return;
    }

    vector<int>shortestPath = graph.TSPsolve();

    if (shortestPath[0] == -1)
    {
        ui->label_9->setText("Невозможно решить задачу коммивояжера");
        return;
    }

    QString resultString;
    for (int i = 0; i < shortestPath.size(); i++)
    {
        resultString.append(QString::number(shortestPath[i]));
        if (i < shortestPath.size() - 1)
        {
            resultString.append(", ");
        }
    }
    ui->label_9->setText(resultString);
```

```
    ui->label_10->setText(QString::number(graph.way(shortestPath)));
    ui->label_11->setText("Путь: ");
    update();
}
```

tsp.ui

```
<?xml version="1.0" encoding="UTF-8"?>
<ui version="4.0">
 <class>TSP</class>
 <widget class="QMainWindow" name="TSP">
  <property name="geometry">
   <rect>
    <x>0</x>
    <y>0</y>
    <width>1307</width>
    <height>661</height>
   </rect>
  </property>
  <property name="windowTitle">
   <string>TSP</string>
  </property>
  <widget class="QWidget" name="centralwidget">
   <widget class="QPushButton" name="pushButton">
    <property name="geometry">
     <rect>
      <x>950</x>
      <y>10</y>
      <width>101</width>
      <height>31</height>
     </rect>
    </property>
    <property name="text">
     <string>Добавить узел</string>
    </property>
   </widget>
   <widget class="QLineEdit" name="lineEdit">
    <property name="geometry">
     <rect>
      <x>830</x>
      <y>10</y>
      <width>113</width>
      <height>31</height>
     </rect>
    </property>
   </widget>
   <widget class="QPushButton" name="pushButton_2">
    <property name="geometry">
     <rect>
      <x>1190</x>
      <y>90</y>
      <width>111</width>
      <height>31</height>
     </rect>
    </property>
    <property name="text">
     <string>Добавить ребро</string>
    </property>
   </widget>
   <widget class="QLineEdit" name="lineEdit_2">
    <property name="geometry">
     <rect>
      <x>830</x>
      <y>90</y>
```

```
    <width>113</width>
    <height>31</height>
   </rect>
  </property>
 </widget>
 <widget class="QLineEdit" name="lineEdit_3">
  <property name="geometry">
   <rect>
    <x>950</x>
    <y>90</y>
    <width>113</width>
    <height>31</height>
   </rect>
  </property>
 </widget>
 <widget class="QLineEdit" name="lineEdit_4">
  <property name="geometry">
   <rect>
    <x>1070</x>
    <y>90</y>
    <width>113</width>
    <height>31</height>
   </rect>
  </property>
 </widget>
 <widget class="QLabel" name="label">
  <property name="geometry">
   <rect>
    <x>860</x>
    <y>120</y>
    <width>55</width>
    <height>16</height>
   </rect>
  </property>
  <property name="text">
   <string>Откуда</string>
  </property>
 </widget>
 <widget class="QLabel" name="label_2">
  <property name="geometry">
   <rect>
    <x>990</x>
    <y>120</y>
    <width>55</width>
    <height>16</height>
   </rect>
  </property>
  <property name="text">
   <string>Куда</string>
  </property>
 </widget>
 <widget class="QLabel" name="label_3">
  <property name="geometry">
   <rect>
    <x>1100</x>
    <y>120</y>
    <width>55</width>
    <height>16</height>
   </rect>
  </property>
  <property name="text">
   <string>Сколько</string>
  </property>
 </widget>
 <widget class="QPushButton" name="pushButton_3">
  <property name="geometry">
```

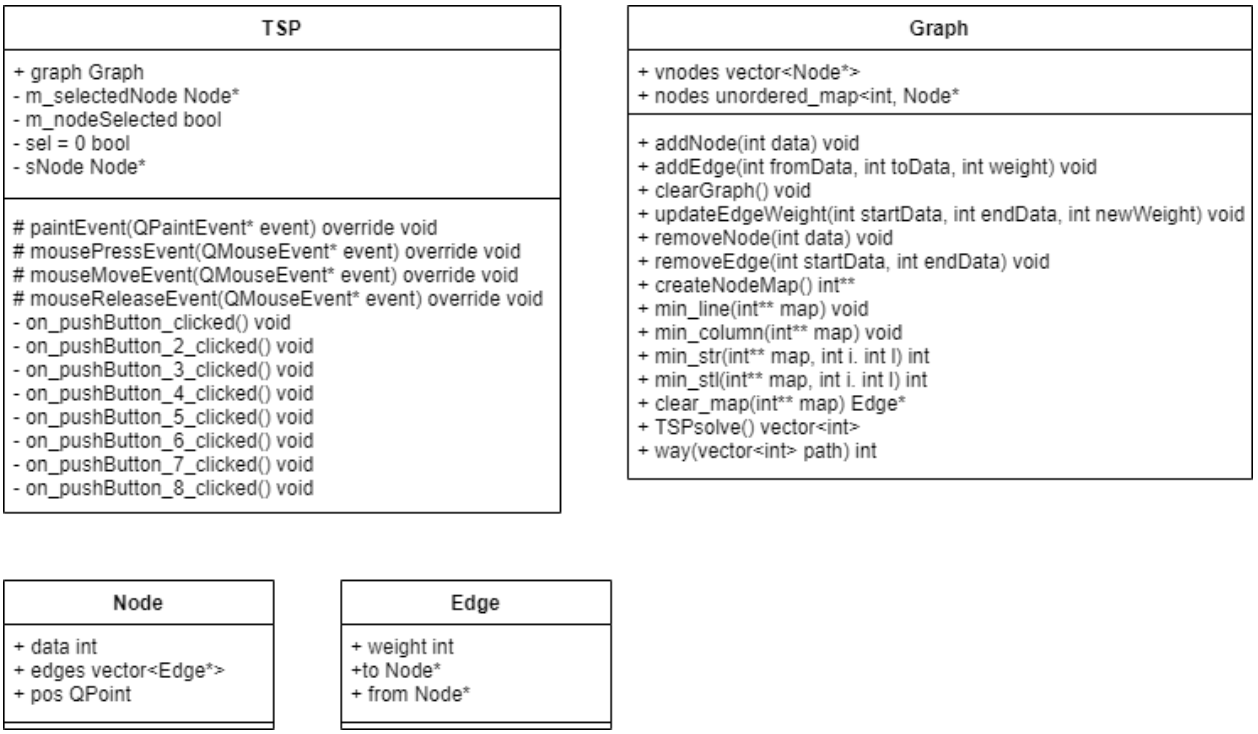
```
  <rect>
   <x>950</x>
   <y>50</y>
   <width>101</width>
   <height>31</height>
  </rect>
 </property>
 <property name="text">
  <string>Удалить узел</string>
 </property>
</widget>
<widget class="QLineEdit" name="lineEdit_5">
 <property name="geometry">
  <rect>
   <x>830</x>
   <y>50</y>
   <width>113</width>
   <height>31</height>
  </rect>
 </property>
</widget>
<widget class="QPushButton" name="pushButton_4">
 <property name="geometry">
  <rect>
   <x>1070</x>
   <y>190</y>
   <width>101</width>
   <height>31</height>
  </rect>
 </property>
 <property name="text">
  <string>Удалить ребро</string>
 </property>
</widget>
<widget class="QLineEdit" name="lineEdit_6">
 <property name="geometry">
  <rect>
   <x>830</x>
   <y>190</y>
   <width>113</width>
   <height>31</height>
  </rect>
 </property>
</widget>
<widget class="QLineEdit" name="lineEdit_7">
 <property name="geometry">
  <rect>
   <x>950</x>
   <y>190</y>
   <width>113</width>
   <height>31</height>
  </rect>
 </property>
</widget>
<widget class="QLabel" name="label_4">
 <property name="geometry">
  <rect>
   <x>860</x>
   <y>220</y>
   <width>55</width>
   <height>16</height>
  </rect>
 </property>
 <property name="text">
  <string>Откуда</string>
 </property>
```

```
</widget>
<widget class="QLabel" name="label_5">
 <property name="geometry">
  <rect>
   <x>990</x>
   <y>220</y>
   <width>55</width>
   <height>16</height>
  </rect>
 </property>
 <property name="text">
  <string>Куда</string>
 </property>
</widget>
<widget class="QPushButton" name="pushButton_5">
 <property name="geometry">
  <rect>
   <x>830</x>
   <y>240</y>
   <width>161</width>
   <height>31</height>
  </rect>
 </property>
 <property name="text">
  <string>Создать граф варианта 2</string>
 </property>
</widget>
<widget class="QPushButton" name="pushButton_6">
 <property name="geometry">
  <rect>
   <x>1190</x>
   <y>140</y>
   <width>111</width>
   <height>31</height>
  </rect>
 </property>
 <property name="text">
  <string>Изменить ребро</string>
 </property>
</widget>
<widget class="QLineEdit" name="lineEdit_8">
 <property name="geometry">
  <rect>
   <x>830</x>
   <y>140</y>
   <width>113</width>
   <height>31</height>
  </rect>
 </property>
</widget>
<widget class="QLineEdit" name="lineEdit_9">
 <property name="geometry">
  <rect>
   <x>950</x>
   <y>140</y>
   <width>113</width>
   <height>31</height>
  </rect>
 </property>
</widget>
<widget class="QLineEdit" name="lineEdit_10">
 <property name="geometry">
  <rect>
   <x>1070</x>
   <y>140</y>
   <width>113</width>
```

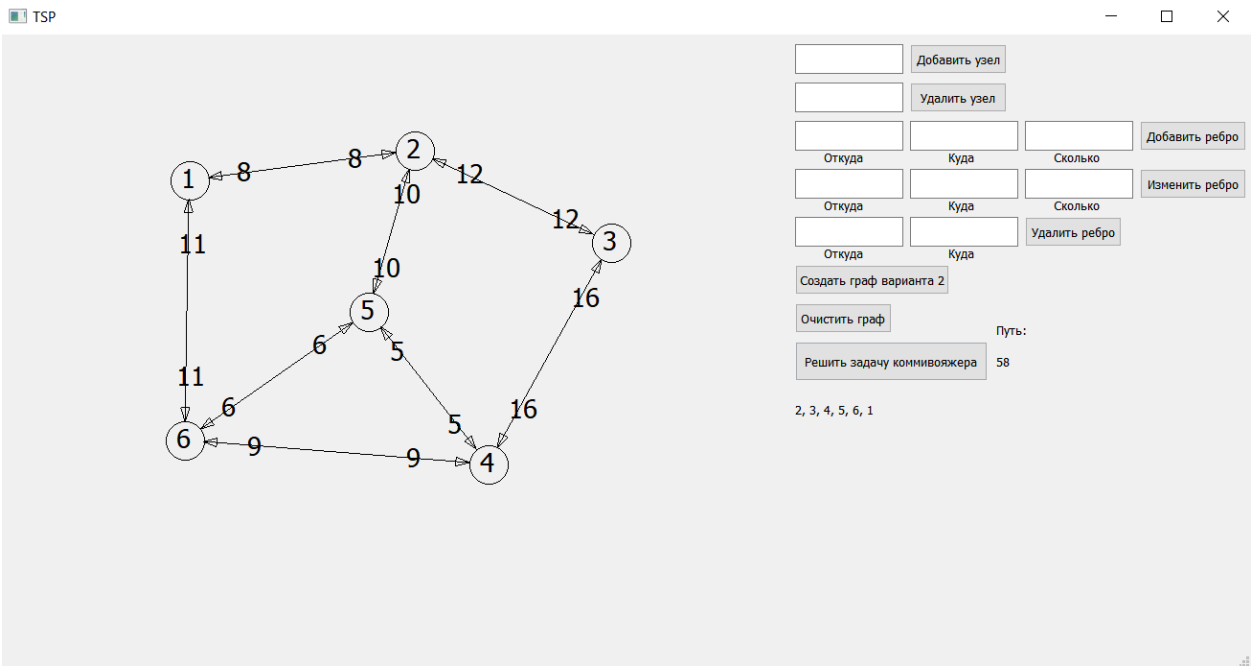
```
    <height>31</height>
   </rect>
  </property>
 </widget>
 <widget class="QLabel" name="label_6">
  <property name="geometry">
   <rect>
    <x>860</x>
    <y>170</y>
    <width>55</width>
    <height>16</height>
   </rect>
  </property>
  <property name="text">
   <string>Откуда</string>
  </property>
 </widget>
 <widget class="QLabel" name="label_7">
  <property name="geometry">
   <rect>
    <x>990</x>
    <y>170</y>
    <width>55</width>
    <height>16</height>
   </rect>
  </property>
  <property name="text">
   <string>Куда</string>
  </property>
 </widget>
 <widget class="QLabel" name="label_8">
  <property name="geometry">
   <rect>
    <x>1100</x>
    <y>170</y>
    <width>55</width>
    <height>16</height>
   </rect>
  </property>
  <property name="text">
   <string>Сколько</string>
  </property>
 </widget>
 <widget class="QPushButton" name="pushButton_7">
  <property name="geometry">
   <rect>
    <x>830</x>
    <y>280</y>
    <width>101</width>
    <height>31</height>
   </rect>
  </property>
  <property name="text">
   <string>Очистить граф</string>
  </property>
 </widget>
 <widget class="QPushButton" name="pushButton_8">
  <property name="geometry">
   <rect>
    <x>830</x>
    <y>320</y>
    <width>201</width>
    <height>41</height>
   </rect>
  </property>
  <property name="text">
```

```
      <string>Решить задачу коммивояжера</string>
     </property>
    </widget>
    <widget class="QLabel" name="label_9">
     <property name="geometry">
      <rect>
       <x>830</x>
       <y>370</y>
       <width>471</width>
       <height>41</height>
      </rect>
     </property>
     <property name="text">
      <string/>
     </property>
    </widget>
    <widget class="QLabel" name="label_10">
     <property name="geometry">
      <rect>
       <x>1040</x>
       <y>320</y>
       <width>251</width>
       <height>41</height>
      </rect>
     </property>
     <property name="text">
      <string/>
     </property>
    </widget>
    <widget class="QLabel" name="label_11">
     <property name="geometry">
      <rect>
       <x>1040</x>
       <y>300</y>
       <width>55</width>
       <height>16</height>
      </rect>
     </property>
     <property name="text">
      <string/>
     </property>
    </widget>
   </widget>
   <widget class="QStatusBar" name="statusbar"/>
  </widget>
  <resources/>
  <connections/>
 </ui>
```

UML – диаграмма



Внешний вид программы:



АРМ специалиста

Тема: Разработка системы для учета плана добычи и отгрузки руды.

Программа содержит следующие элементы:

- Авторизация с помощью пароля
- Доступ к таблицам
- Работа с таблицами: сохранение/загрузка таблицы, добавление строки, очищение пустых строк.

- Доп. функциональные кнопки в таблице «План» для расчета суточной нормы.

Код на c++

MiningIndustry.pro

```
QT       += core gui

greaterThan(QT_MAJOR_VERSION, 4): QT += widgets

CONFIG += c++11

# You can make your code fail to compile if it uses deprecated APIs.
# In order to do so, uncomment the following line.
#DEFINES += QT_DISABLE_DEPRECATED_BEFORE=0x060000    # disables all the APIs
deprecated before Qt 6.0.0

SOURCES += \
    main.cpp \
    mainwindow.cpp \
    menu.cpp \
    password.cpp \
    plan.cpp \
    ship.cpp

HEADERS += \
    mainwindow.h \
    menu.h \
    password.h \
    plan.h \
    ship.h

FORMS += \
    mainwindow.ui \
    menu.ui \
    password.ui \
    plan.ui \
    ship.ui

# Default rules for deployment.
qnx: target.path = /tmp/$${TARGET}/bin
else: unix:!android: target.path = /opt/$${TARGET}/bin
!isEmpty(target.path): INSTALLS += target
```

mainwindow.h

```
#ifndef MAINWINDOW_H
#define MAINWINDOW_H
#include <QMainWindow>
#include <QDialog>

QT_BEGIN_NAMESPACE
namespace Ui { class MainWindow; }
QT_END_NAMESPACE

class MainWindow : public QMainWindow
{
    Q_OBJECT

public:
    MainWindow(QWidget *parent = nullptr);
```

```
    ~MainWindow();

private slots:
    void on_enterButton_clicked();

    void on_saveButton_clicked();

    void on_newButton_clicked();

    void on_clearButton_clicked();

    void on_quitButton_clicked();

    void on_delButton_clicked();

private:
    Ui::MainWindow *ui;
};
#endif // MAINWINDOW_H
```

menu.h

```
#ifndef MENU_H
#define MENU_H

#include <QDialog>
#include "mainwindow.h"
#include "plan.h"
#include "ship.h"

namespace Ui {
class Menu;
}

class Menu : public QDialog
{
    Q_OBJECT

public:
    explicit Menu(QWidget *parent = nullptr);
    ~Menu();

private slots:
    void on_miningButton_clicked();

    void on_quitButton_clicked();

    void on_planButton_clicked();

    void on_shipButton_clicked();

private:
    Ui::Menu *ui;
    MainWindow *window;
    Plan *pwindow;
    Ship *swindow;
};

#endif // MENU_H
```

password.h

```
#ifndef PASSWORD_H
```

```
#define PASSWORD_H

#include <QDialog>
#include <QMainWindow>
#include "menu.h"

namespace Ui {
class Password;
}

class Password : public QDialog
{
    Q_OBJECT

public:
    explicit Password(QWidget *parent = nullptr);
    ~Password();

private slots:
    void on_enterButton_clicked();

    void on_quitButton_clicked();

private:
    Ui::Password *ui;
    Menu *window;
};

#endif // PASSWORD_H
```

plan.h

```
#ifndef PLAN_H
#define PLAN_H

#include <QDialog>

namespace Ui {
class Plan;
}

class Plan : public QDialog
{
    Q_OBJECT

public:
    explicit Plan(QWidget *parent = nullptr);
    ~Plan();

private slots:
    void on_quitButton_clicked();

    void on_enterButton_clicked();

    void on_saveButton_clicked();

    void on_newButton_clicked();

    void on_clearButton_clicked();

    void on_delButton_clicked();

    void on_count31Button_clicked();
```

```
    void on_count30Button_clicked();

    void on_count29Button_clicked();

    void on_count28Button_clicked();

private:
    Ui::Plan *ui;
};

#endif // PLAN_H
```

ship.h

```
#ifndef SHIP_H
#define SHIP_H

#include <QDialog>

namespace Ui {
class Ship;
}

class Ship : public QDialog
{
    Q_OBJECT

public:
    explicit Ship(QWidget *parent = nullptr);
    ~Ship();

private slots:
    void on_quitButton_clicked();

    void on_enterButton_clicked();

    void on_saveButton_clicked();

    void on_newButton_clicked();

    void on_clearButton_clicked();

    void on_delButton_clicked();
private:
    Ui::Ship *ui;
};

#endif // SHIP_H
```

main.cpp

```
#include "mainwindow.h"
#include "password.h"

#include <QApplication>

int main(int argc, char *argv[])
{
    QApplication a(argc, argv);
    Password w;
    w.show();
    return a.exec();
}
```

mainwindow.cpp

```
#include "mainwindow.h"
#include "ui_mainwindow.h"
#include <QFile>
#include <QTextStream>

MainWindow::MainWindow(QWidget *parent)
    : QMainWindow(parent)
    , ui(new Ui::MainWindow)
{
    ui->setupUi(this);
}

MainWindow::~MainWindow()
{
    delete ui;
}

void MainWindow::on_clearButton_clicked()
{
    ui->tableWidget->setColumnCount(5);
    ui->tableWidget->setHorizontalHeaderLabels({"Дата", "Участок", "Бригада",
"Смена", "Объем"});

    for (int row = 0; row < ui->tableWidget->rowCount(); ++row)
    {
        bool isEmpty = true;
        for (int col = 0; col < ui->tableWidget->columnCount(); ++col)
        {
            if (ui->tableWidget->item(row, col) && !ui->tableWidget-
>item(row, col)->text().isEmpty())
            {
                isEmpty = false;
                break;
            }
        }
        if (isEmpty)
        {
            ui->tableWidget->removeRow(row);
            row--;
        }
    }

    ui->label->setText("Пустые ряды удалены");
}

void MainWindow::on_enterButton_clicked()
{
    ui->tableWidget->setColumnCount(5);
    ui->tableWidget->setHorizontalHeaderLabels({"Дата", "Участок", "Бригада",
"Смена", "Объем"});
    ui->tableWidget->setRowCount(0);

    QFile file("mining.txt");
    if (!file.open(QIODevice::ReadOnly | QIODevice::Text))
    {
        ui->label->setText("Файл не найден, создан новый");
        QFile file("mining.txt");
        file.open(QIODevice::WriteOnly);
        file.close();
        return;
    }
```

```
    int row = 0;
    QTextStream in(&file);
    while (!in.atEnd())
    {
        QString line = in.readLine();
        QStringList fields = line.split("\t");

        ui->tableWidget->insertRow(row);
        ui->tableWidget->setColumnCount(fields.size());
        for (int col = 0; col < 5; ++col)
        {
            ui->tableWidget->setItem(row, col, new
QTableWidgetItem(fields[col]));
        }

        ++row;
    }

    file.close();

    ui->label->setText("Таблица загружена");
}


void MainWindow::on_newButton_clicked()
{
    int row = ui->tableWidget->rowCount();
    ui->tableWidget->insertRow(row);
    for (int col = 0; col < 5; ++col)
    {
        ui->tableWidget->setItem(row, col, new QTableWidgetItem(""));
    }
    ui->label->setText("");
}

void MainWindow::on_saveButton_clicked()
{
    on_clearButton_clicked();

    QFile file("mining.txt");
    file.open(QIODevice::WriteOnly | QIODevice::Text);

    QTextStream out(&file);
    out.setCodec("UTF-8");
    for (int row = 0; row < ui->tableWidget->rowCount(); ++row)
    {
        for (int col = 0; col < 5; ++col)
        {
            if(ui->tableWidget->item(row, col) != nullptr)
            {
                out << ui->tableWidget->item(row, col)->text();
            }
            if (col != ui->tableWidget->columnCount() - 1){out << "\t";}
        }
        if (row != ui->tableWidget->rowCount() - 1){out << '\n';}
    }
    file.close();

    ui->label->setText("Таблица сохранена");
}

void MainWindow::on_quitButton_clicked()
{
    this->close();
}
```

```
void MainWindow::on_delButton_clicked()
{
    int row = ui->lineEdit->text().toInt();
    if (row > 0 && row <= ui->tableWidget->rowCount())
    {
        ui->tableWidget->removeRow(row - 1);
        ui->label->setText("");
    }
    else
    {
        ui->label->setText("Такого ряда нет");
    }
    ui->lineEdit->clear();
}
```

menu.cpp

```
#include "menu.h"
#include "ui_menu.h"

Menu::Menu(QWidget *parent) :
    QDialog(parent),
    ui(new Ui::Menu)
{
    ui->setupUi(this);
}

Menu::~Menu()
{
    delete ui;
}

void Menu::on_miningButton_clicked()
{
    window = new MainWindow(this);
    window->show();
}

void Menu::on_quitButton_clicked()
{
    exit(0);
}


void Menu::on_planButton_clicked()
{
    pwindow = new Plan(this);
    pwindow->show();
}


void Menu::on_shipButton_clicked()
{
    swindow = new Ship(this);
    swindow->show();
}
```

password.cpp

```
#include "password.h"
#include "ui_password.h"
```

```
Password::Password(QWidget *parent) :
    QDialog(parent),
    ui(new Ui::Password)
{
    ui->setupUi(this);
}

Password::~Password()
{
    delete ui;
}

void Password::on_enterButton_clicked()
{
    QString line = ui->passwordLine->text();
    if (line == "password")
    {
        this->close();
        window = new Menu(this);
        window->show();
    }
    else
    {
        ui->label_2->setText("Неверно");
    }
}


void Password::on_quitButton_clicked()
{
    exit(0);
}
```

plan.cpp

```
#include "plan.h"
#include "ui_plan.h"
#include <QFile>
#include <QTextStream>

Plan::Plan(QWidget *parent) :
    QDialog(parent),
    ui(new Ui::Plan)
{
    ui->setupUi(this);
}

Plan::~Plan()
{
    delete ui;
}

void Plan::on_quitButton_clicked()
{
    this->close();
}

void Plan::on_clearButton_clicked()
{
    ui->tableWidget->setColumnCount(3);
    ui->tableWidget->setHorizontalHeaderLabels({"Участок", "На месяц", "На
сутки"});
```

```
    for (int row = 0; row < ui->tableWidget->rowCount(); ++row)
    {
        bool isEmpty = true;
        for (int col = 0; col < ui->tableWidget->columnCount(); ++col)
        {
            if (ui->tableWidget->item(row, col) && !ui->tableWidget-
>item(row, col)->text().isEmpty())
            {
                isEmpty = false;
                break;
            }
        }
        if (isEmpty)
        {
            ui->tableWidget->removeRow(row);
            row--;
        }
    }

    ui->label->setText("Пустые ряды удалены");
}

void Plan::on_enterButton_clicked()
{
    ui->tableWidget->setColumnCount(3);
    ui->tableWidget->setHorizontalHeaderLabels({"Участок", "На месяц", "На
сутки"});
    ui->tableWidget->setRowCount(0);

    QFile file("plan.txt");
    if (!file.open(QIODevice::ReadOnly | QIODevice::Text))
    {
        ui->label->setText("Файл не найден, создан новый");
        QFile file("plan.txt");
        file.open(QIODevice::WriteOnly);
        file.close();
        return;
    }

    int row = 0;
    QTextStream in(&file);
    while (!in.atEnd())
    {
        QString line = in.readLine();
        QStringList fields = line.split("\t");

        ui->tableWidget->insertRow(row);
        ui->tableWidget->setColumnCount(fields.size());
        for (int col = 0; col < 3; ++col)
        {
            ui->tableWidget->setItem(row, col, new
QTableWidgetItem(fields[col]));
        }

        ++row;
    }

    file.close();

    ui->label->setText("Таблица загружена");
}


void Plan::on_newButton_clicked()
{
    int row = ui->tableWidget->rowCount();
```

```
    ui->tableWidget->insertRow(row);
    for (int col = 0; col < 3; ++col)
    {
        ui->tableWidget->setItem(row, col, new QTableWidgetItem(""));
    }
    ui->label->setText("");
}

void Plan::on_saveButton_clicked()
{
    on_clearButton_clicked();

    QFile file("plan.txt");
    file.open(QIODevice::WriteOnly | QIODevice::Text);

    QTextStream out(&file);
    out.setCodec("UTF-8");
    for (int row = 0; row < ui->tableWidget->rowCount(); ++row)
    {
        for (int col = 0; col < 3; ++col)
        {
            if(ui->tableWidget->item(row, col) != nullptr)
            {
                out << ui->tableWidget->item(row, col)->text();
            }
            if (col != ui->tableWidget->columnCount() - 1){out << "\t";}
        }
        if (row != ui->tableWidget->rowCount() - 1){out << '\n';}
    }
    file.close();

    ui->label->setText("Таблица сохранена");
}

void Plan::on_delButton_clicked()
{
    int row = ui->lineEdit->text().toInt();
    if (row > 0 && row <= ui->tableWidget->rowCount())
    {
        ui->tableWidget->removeRow(row - 1);
        ui->label->setText("");
    }
    else
    {
        ui->label->setText("Такого ряда нет");
    }
    ui->lineEdit->clear();
}

void Plan::on_count31Button_clicked()
{
    for (int i = 0; i < ui->tableWidget->rowCount(); i++)
    {
        int count = ui->tableWidget->item(i,1)->text().toInt()/31;
        QTableWidgetItem *item = new
QTableWidgetItem(QString::number(count));
        ui->tableWidget->setItem(i, 2, item);
    }
}


void Plan::on_count30Button_clicked()
{
    for (int i = 0; i < ui->tableWidget->rowCount(); i++)
    {
        int count = ui->tableWidget->item(i,1)->text().toInt()/30;
```

```
        QTableWidgetItem *item = new
QTableWidgetItem(QString::number(count));
        ui->tableWidget->setItem(i, 2, item);
    }
}


void Plan::on_count29Button_clicked()
{
    for (int i = 0; i < ui->tableWidget->rowCount(); i++)
    {
        int count = ui->tableWidget->item(i,1)->text().toInt()/29;
        QTableWidgetItem *item = new
QTableWidgetItem(QString::number(count));
        ui->tableWidget->setItem(i, 2, item);
    }
}


void Plan::on_count28Button_clicked()
{
    for (int i = 0; i < ui->tableWidget->rowCount(); i++)
    {
        int count = ui->tableWidget->item(i,1)->text().toInt()/28;
        QTableWidgetItem *item = new
QTableWidgetItem(QString::number(count));
        ui->tableWidget->setItem(i, 2, item);
    }
}
```

ship.cpp

```
#include "ship.h"
#include "ui_ship.h"
#include <QFile>
#include <QTextStream>

Ship::Ship(QWidget *parent) :
    QDialog(parent),
    ui(new Ui::Ship)
{
    ui->setupUi(this);
}

Ship::~Ship()
{
    delete ui;
}

void Ship::on_quitButton_clicked()
{
    this->close();
}

void Ship::on_clearButton_clicked()
{
    ui->tableWidget->setColumnCount(6);
    ui->tableWidget->setHorizontalHeaderLabels({"Отгрузка", "Поступление",
"Накладные", "Вагоны", "Объем", "Заказчик"});

    for (int row = 0; row < ui->tableWidget->rowCount(); ++row)
    {
        bool isEmpty = true;
        for (int col = 0; col < ui->tableWidget->columnCount(); ++col)
```

```
        {
            if (ui->tableWidget->item(row, col) && !ui->tableWidget-
>item(row, col)->text().isEmpty())
            {
                isEmpty = false;
                break;
            }
        }
        if (isEmpty)
        {
            ui->tableWidget->removeRow(row);
            row--;
        }
    }

    ui->label->setText("Пустые ряды удалены");
}

void Ship::on_enterButton_clicked()
{
    ui->tableWidget->setColumnCount(6);
    ui->tableWidget->setHorizontalHeaderLabels({"Отгрузка", "Поступление",
"Накладные", "Вагоны", "Объем", "Заказчик"});
    ui->tableWidget->setRowCount(0);

    QFile file("ship.txt");
    if (!file.open(QIODevice::ReadOnly | QIODevice::Text))
    {
        ui->label->setText("Файл не найден, создан новый");
        QFile file("ship.txt");
        file.open(QIODevice::WriteOnly);
        file.close();
        return;
    }

    int row = 0;
    QTextStream in(&file);
    while (!in.atEnd())
    {
        QString line = in.readLine();
        QStringList fields = line.split("\t");

        ui->tableWidget->insertRow(row);
        ui->tableWidget->setColumnCount(fields.size());
        for (int col = 0; col < 6; ++col)
        {
            ui->tableWidget->setItem(row, col, new
QTableWidgetItem(fields[col]));
        }

        ++row;
    }

    file.close();

    ui->label->setText("Таблица загружена");
}


void Ship::on_newButton_clicked()
{
    int row = ui->tableWidget->rowCount();
    ui->tableWidget->insertRow(row);
    for (int col = 0; col < 6; ++col)
    {
        ui->tableWidget->setItem(row, col, new QTableWidgetItem(""));
```

```
    }
    ui->label->setText("");
}

void Ship::on_saveButton_clicked()
{
    on_clearButton_clicked();

    QFile file("ship.txt");
    file.open(QIODevice::WriteOnly | QIODevice::Text);

    QTextStream out(&file);
    out.setCodec("UTF-8");
    for (int row = 0; row < ui->tableWidget->rowCount(); ++row)
    {
        for (int col = 0; col < 6; ++col)
        {
            if(ui->tableWidget->item(row, col) != nullptr)
            {
                out << ui->tableWidget->item(row, col)->text();
            }
            if (col != ui->tableWidget->columnCount() - 1){out << "\t";}
        }
        if (row != ui->tableWidget->rowCount() - 1){out << '\n';}
    }
    file.close();

    ui->label->setText("Таблица сохранена");
}

void Ship::on_delButton_clicked()
{
    int row = ui->lineEdit->text().toInt();
    if (row > 0 && row <= ui->tableWidget->rowCount())
    {
        ui->tableWidget->removeRow(row - 1);
        ui->label->setText("");
    }
    else
    {
        ui->label->setText("Такого ряда нет");
    }
    ui->lineEdit->clear();
}
```

mainwindow.ui

```
<?xml version="1.0" encoding="UTF-8"?>
<ui version="4.0">
 <class>MainWindow</class>
 <widget class="QMainWindow" name="MainWindow">
  <property name="geometry">
   <rect>
    <x>0</x>
    <y>0</y>
    <width>789</width>
    <height>508</height>
   </rect>
  </property>
  <property name="windowTitle">
   <string>Добыча</string>
  </property>
  <widget class="QWidget" name="centralwidget">
   <widget class="QTableWidget" name="tableWidget">
```

```
 <property name="geometry">
  <rect>
   <x>10</x>
   <y>50</y>
   <width>651</width>
   <height>581</height>
  </rect>
 </property>
 <column>
  <property name="text">
   <string>Дата</string>
  </property>
 </column>
 <column>
  <property name="text">
   <string>Участок</string>
  </property>
 </column>
 <column>
  <property name="text">
   <string>Бригада</string>
  </property>
 </column>
 <column>
  <property name="text">
   <string>Смена</string>
  </property>
 </column>
 <column>
  <property name="text">
   <string>Объем</string>
  </property>
 </column>
</widget>
<widget class="QLabel" name="label">
 <property name="geometry">
  <rect>
   <x>590</x>
   <y>10</y>
   <width>181</width>
   <height>31</height>
  </rect>
 </property>
 <property name="text">
  <string/>
 </property>
</widget>
<widget class="QPushButton" name="quitButton">
 <property name="geometry">
  <rect>
   <x>680</x>
   <y>50</y>
   <width>93</width>
   <height>31</height>
  </rect>
 </property>
 <property name="text">
  <string>Выход</string>
 </property>
</widget>
<widget class="QWidget" name="layoutWidget">
 <property name="geometry">
  <rect>
   <x>10</x>
   <y>10</y>
   <width>571</width>
```

```
      <height>30</height>
     </rect>
    </property>
    <layout class="QHBoxLayout" name="horizontalLayout">
     <item>
      <widget class="QPushButton" name="enterButton">
       <property name="text">
        <string>Вывести</string>
       </property>
      </widget>
     </item>
     <item>
      <widget class="QPushButton" name="saveButton">
       <property name="text">
        <string>Сохранить</string>
       </property>
      </widget>
     </item>
     <item>
      <widget class="QPushButton" name="clearButton">
       <property name="text">
        <string>Очистить</string>
       </property>
      </widget>
     </item>
     <item>
      <widget class="QPushButton" name="newButton">
       <property name="text">
        <string>Ввести</string>
       </property>
      </widget>
     </item>
     <item>
      <widget class="QPushButton" name="delButton">
       <property name="text">
        <string>Удалить ряд:</string>
       </property>
      </widget>
     </item>
     <item>
      <widget class="QLineEdit" name="lineEdit"/>
     </item>
    </layout>
   </widget>
  </widget>
  <widget class="QStatusBar" name="statusbar"/>
 </widget>
 <resources/>
 <connections/>
</ui>
```

menu.ui

```
<?xml version="1.0" encoding="UTF-8"?>
<ui version="4.0">
 <class>Menu</class>
 <widget class="QDialog" name="Menu">
  <property name="geometry">
   <rect>
    <x>0</x>
    <y>0</y>
    <width>917</width>
    <height>45</height>
   </rect>
```

```
  </property>
  <property name="windowTitle">
   <string>Меню</string>
  </property>
  <widget class="QPushButton" name="miningButton">
   <property name="geometry">
    <rect>
     <x>10</x>
     <y>10</y>
     <width>93</width>
     <height>28</height>
    </rect>
   </property>
   <property name="text">
    <string>Добыча</string>
   </property>
  </widget>
  <widget class="QPushButton" name="quitButton">
   <property name="geometry">
    <rect>
     <x>810</x>
     <y>10</y>
     <width>93</width>
     <height>28</height>
    </rect>
   </property>
   <property name="text">
    <string>Выход</string>
   </property>
  </widget>
  <widget class="QPushButton" name="planButton">
   <property name="geometry">
    <rect>
     <x>110</x>
     <y>10</y>
     <width>93</width>
     <height>28</height>
    </rect>
   </property>
   <property name="text">
    <string>План</string>
   </property>
  </widget>
  <widget class="QPushButton" name="shipButton">
   <property name="geometry">
    <rect>
     <x>210</x>
     <y>10</y>
     <width>93</width>
     <height>28</height>
    </rect>
   </property>
   <property name="text">
    <string>Отгрузка</string>
   </property>
  </widget>
 </widget>
 <resources/>
 <connections/>
</ui>
```

password.ui

```
<?xml version="1.0" encoding="UTF-8"?>
```

```
<ui version="4.0">
 <class>Password</class>
 <widget class="QDialog" name="Password">
  <property name="geometry">
   <rect>
    <x>0</x>
    <y>0</y>
    <width>685</width>
    <height>509</height>
   </rect>
  </property>
  <property name="windowTitle">
   <string>Dialog</string>
  </property>
  <widget class="QLabel" name="label">
   <property name="geometry">
    <rect>
     <x>230</x>
     <y>170</y>
     <width>201</width>
     <height>41</height>
    </rect>
   </property>
   <property name="text">
    <string>&lt;html&gt;&lt;head/&gt;&lt;body&gt;&lt;p&gt;&lt;span
style=&quot; font-size:14pt;&quot;&gt;Введите
password:&lt;/span&gt;&lt;/p&gt;&lt;/body&gt;&lt;/html&gt;</string>
   </property>
  </widget>
  <widget class="QLineEdit" name="passwordLine">
   <property name="geometry">
    <rect>
     <x>260</x>
     <y>220</y>
     <width>141</width>
     <height>31</height>
    </rect>
   </property>
  </widget>
  <widget class="QPushButton" name="enterButton">
   <property name="geometry">
    <rect>
     <x>280</x>
     <y>290</y>
     <width>101</width>
     <height>28</height>
    </rect>
   </property>
   <property name="text">
    <string>Ввести</string>
   </property>
  </widget>
  <widget class="QPushButton" name="quitButton">
   <property name="geometry">
    <rect>
     <x>280</x>
     <y>330</y>
     <width>101</width>
     <height>28</height>
    </rect>
   </property>
   <property name="text">
    <string>Выйти</string>
   </property>
  </widget>
  <widget class="QLabel" name="label_2">
```

```
   <property name="geometry">
    <rect>
     <x>300</x>
     <y>260</y>
     <width>81</width>
     <height>20</height>
    </rect>
   </property>
   <property name="text">
    <string/>
   </property>
  </widget>
 </widget>
 <resources/>
 <connections/>
</ui>
```

plan.ui

```
<?xml version="1.0" encoding="UTF-8"?>
<ui version="4.0">
 <class>Plan</class>
 <widget class="QDialog" name="Plan">
  <property name="geometry">
   <rect>
    <x>0</x>
    <y>0</y>
    <width>825</width>
    <height>497</height>
   </rect>
  </property>
  <property name="windowTitle">
   <string>План</string>
  </property>
  <widget class="QPushButton" name="quitButton">
   <property name="geometry">
    <rect>
     <x>720</x>
     <y>10</y>
     <width>93</width>
     <height>28</height>
    </rect>
   </property>
   <property name="text">
    <string>Выход</string>
   </property>
  </widget>
  <widget class="QTableWidget" name="tableWidget">
   <property name="geometry">
    <rect>
     <x>10</x>
     <y>50</y>
     <width>381</width>
     <height>851</height>
    </rect>
   </property>
   <column>
    <property name="text">
     <string>Участок</string>
    </property>
   </column>
   <column>
    <property name="text">
     <string>На месяц</string>
```

```
  </property>
 </column>
 <column>
  <property name="text">
   <string>На сутки</string>
  </property>
 </column>
</widget>
<widget class="QWidget" name="layoutWidget">
 <property name="geometry">
  <rect>
   <x>10</x>
   <y>10</y>
   <width>571</width>
   <height>30</height>
  </rect>
 </property>
 <layout class="QHBoxLayout" name="horizontalLayout_2">
  <item>
   <widget class="QPushButton" name="enterButton">
    <property name="text">
     <string>Вывести</string>
    </property>
   </widget>
  </item>
  <item>
   <widget class="QPushButton" name="saveButton">
    <property name="text">
     <string>Сохранить</string>
    </property>
   </widget>
  </item>
  <item>
   <widget class="QPushButton" name="clearButton">
    <property name="text">
     <string>Очистить</string>
    </property>
   </widget>
  </item>
  <item>
   <widget class="QPushButton" name="newButton">
    <property name="text">
     <string>Ввести</string>
    </property>
   </widget>
  </item>
  <item>
   <widget class="QPushButton" name="delButton">
    <property name="text">
     <string>Удалить ряд:</string>
    </property>
   </widget>
  </item>
  <item>
   <widget class="QLineEdit" name="lineEdit"/>
  </item>
 </layout>
</widget>
<widget class="QLabel" name="label">
 <property name="geometry">
  <rect>
   <x>590</x>
   <y>10</y>
   <width>131</width>
   <height>31</height>
  </rect>
```

```
 </property>
 <property name="text">
  <string/>
 </property>
</widget>
<widget class="QPushButton" name="count31Button">
 <property name="geometry">
  <rect>
   <x>440</x>
   <y>50</y>
   <width>101</width>
   <height>28</height>
  </rect>
 </property>
 <property name="text">
  <string>Рассчитать /31</string>
 </property>
</widget>
<widget class="QLabel" name="label_2">
 <property name="geometry">
  <rect>
   <x>550</x>
   <y>50</y>
   <width>261</width>
   <height>31</height>
  </rect>
 </property>
 <property name="text">
  <string>- вывести план за день по плану за месяц</string>
 </property>
</widget>
<widget class="QPushButton" name="count30Button">
 <property name="geometry">
  <rect>
   <x>440</x>
   <y>90</y>
   <width>101</width>
   <height>28</height>
  </rect>
 </property>
 <property name="text">
  <string>Рассчитать /30</string>
 </property>
</widget>
<widget class="QPushButton" name="count29Button">
 <property name="geometry">
  <rect>
   <x>440</x>
   <y>130</y>
   <width>101</width>
   <height>28</height>
  </rect>
 </property>
 <property name="text">
  <string>Рассчитать /29</string>
 </property>
</widget>
<widget class="QPushButton" name="count28Button">
 <property name="geometry">
  <rect>
   <x>440</x>
   <y>170</y>
   <width>101</width>
   <height>28</height>
  </rect>
 </property>
```

```
   <property name="text">
    <string>Рассчитать /28</string>
   </property>
  </widget>
 </widget>
 <resources/>
 <connections/>
</ui>
```

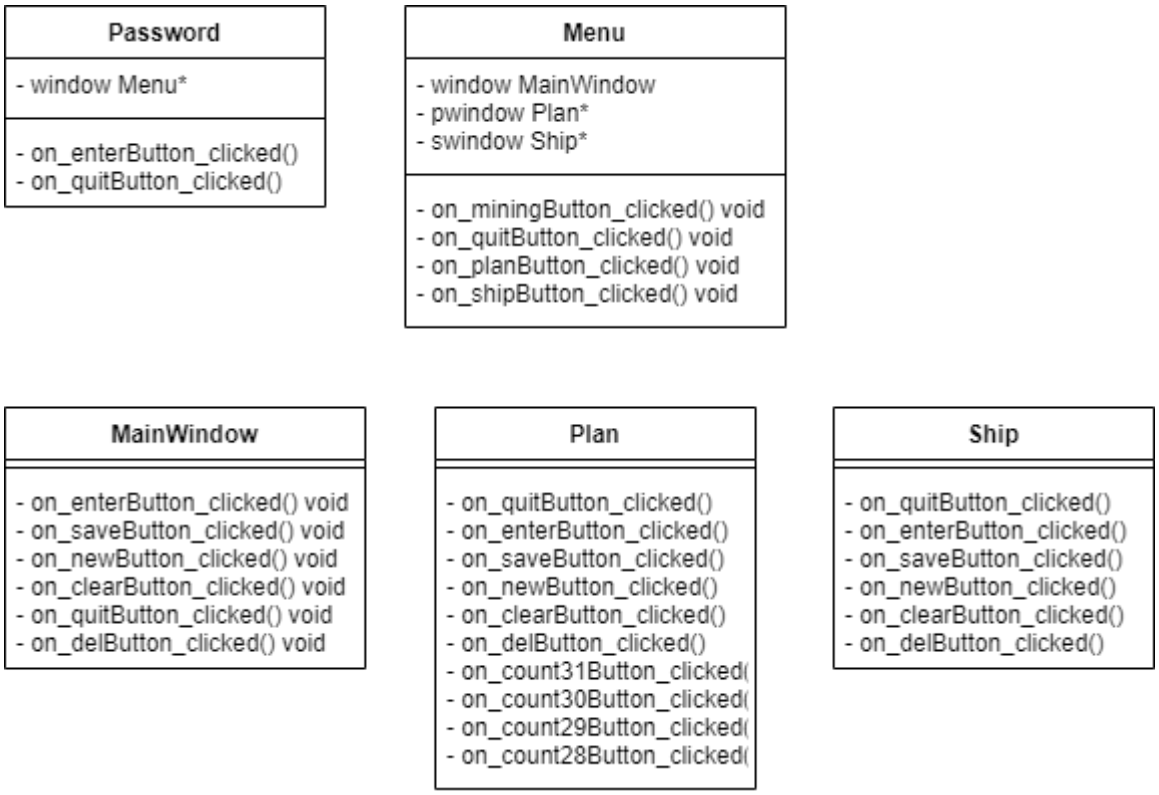
ship.ui

```
<?xml version="1.0" encoding="UTF-8"?>
<ui version="4.0">
 <class>Ship</class>
 <widget class="QDialog" name="Ship">
  <property name="geometry">
   <rect>
    <x>0</x>
    <y>0</y>
    <width>815</width>
    <height>520</height>
   </rect>
  </property>
  <property name="windowTitle">
   <string>Отгрузка</string>
  </property>
  <widget class="QTableWidget" name="tableWidget">
   <property name="geometry">
    <rect>
     <x>10</x>
     <y>50</y>
     <width>781</width>
     <height>931</height>
    </rect>
   </property>
   <column>
    <property name="text">
     <string>Отгрузка</string>
    </property>
   </column>
   <column>
    <property name="text">
     <string>Поступление</string>
    </property>
   </column>
   <column>
    <property name="text">
     <string>Накладные</string>
    </property>
   </column>
   <column>
    <property name="text">
     <string>Вагоны</string>
    </property>
   </column>
   <column>
    <property name="text">
     <string>Объем</string>
    </property>
   </column>
   <column>
    <property name="text">
     <string>Заказчик</string>
    </property>
   </column>
```

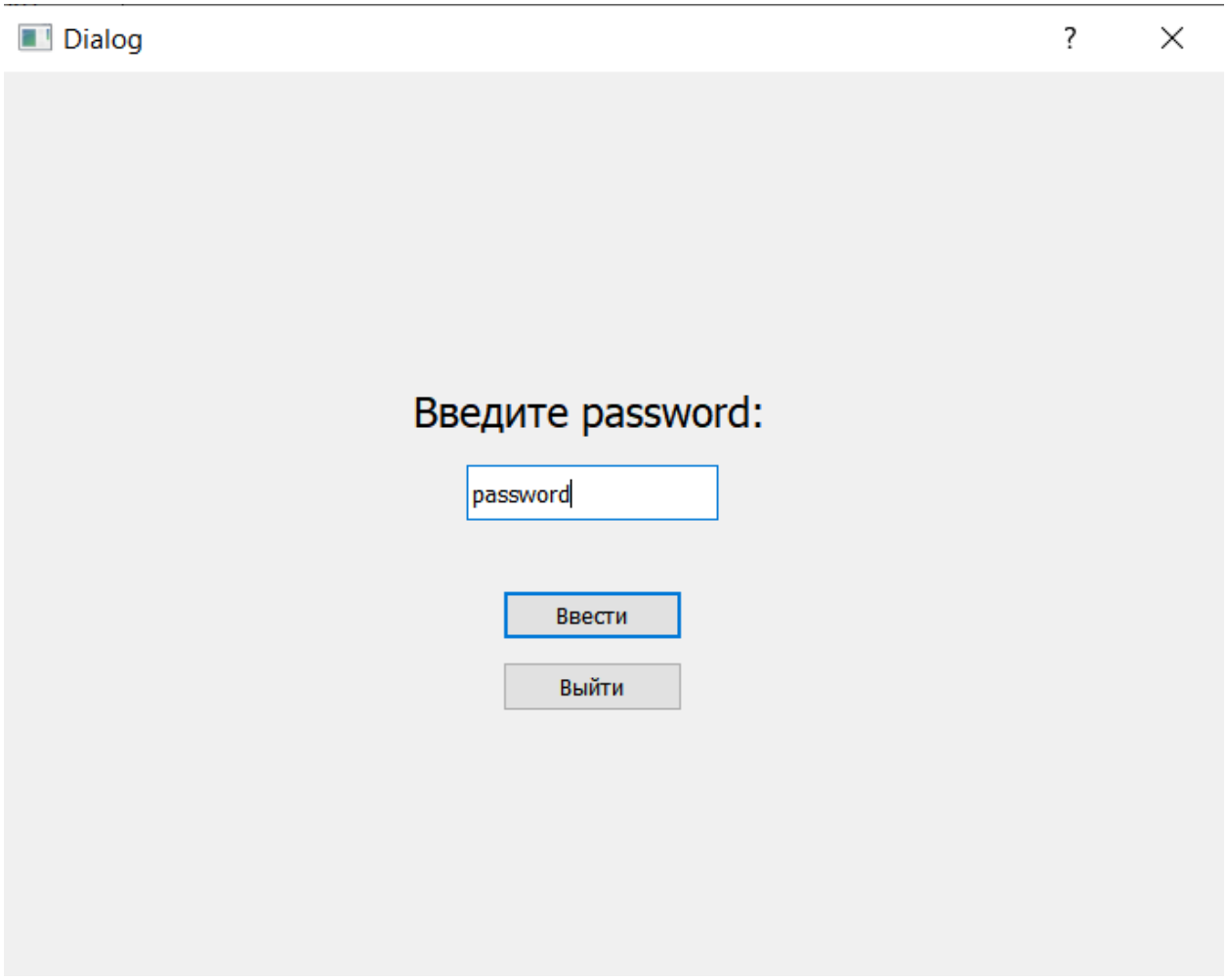
```
</widget>
<widget class="QWidget" name="layoutWidget">
 <property name="geometry">
  <rect>
   <x>10</x>
   <y>10</y>
   <width>571</width>
   <height>30</height>
  </rect>
 </property>
 <layout class="QHBoxLayout" name="horizontalLayout_2">
  <item>
   <widget class="QPushButton" name="enterButton">
    <property name="text">
     <string>Вывести</string>
    </property>
   </widget>
  </item>
  <item>
   <widget class="QPushButton" name="saveButton">
    <property name="text">
     <string>Сохранить</string>
    </property>
   </widget>
  </item>
  <item>
   <widget class="QPushButton" name="clearButton">
    <property name="text">
     <string>Очистить</string>
    </property>
   </widget>
  </item>
  <item>
   <widget class="QPushButton" name="newButton">
    <property name="text">
     <string>Ввести</string>
    </property>
   </widget>
  </item>
  <item>
   <widget class="QPushButton" name="delButton">
    <property name="text">
     <string>Удалить ряд:</string>
    </property>
   </widget>
  </item>
  <item>
   <widget class="QLineEdit" name="lineEdit"/>
  </item>
 </layout>
</widget>
<widget class="QLabel" name="label">
 <property name="geometry">
  <rect>
   <x>590</x>
   <y>10</y>
   <width>131</width>
   <height>31</height>
  </rect>
 </property>
 <property name="text">
  <string/>
 </property>
</widget>
<widget class="QPushButton" name="quitButton">
 <property name="geometry">
```

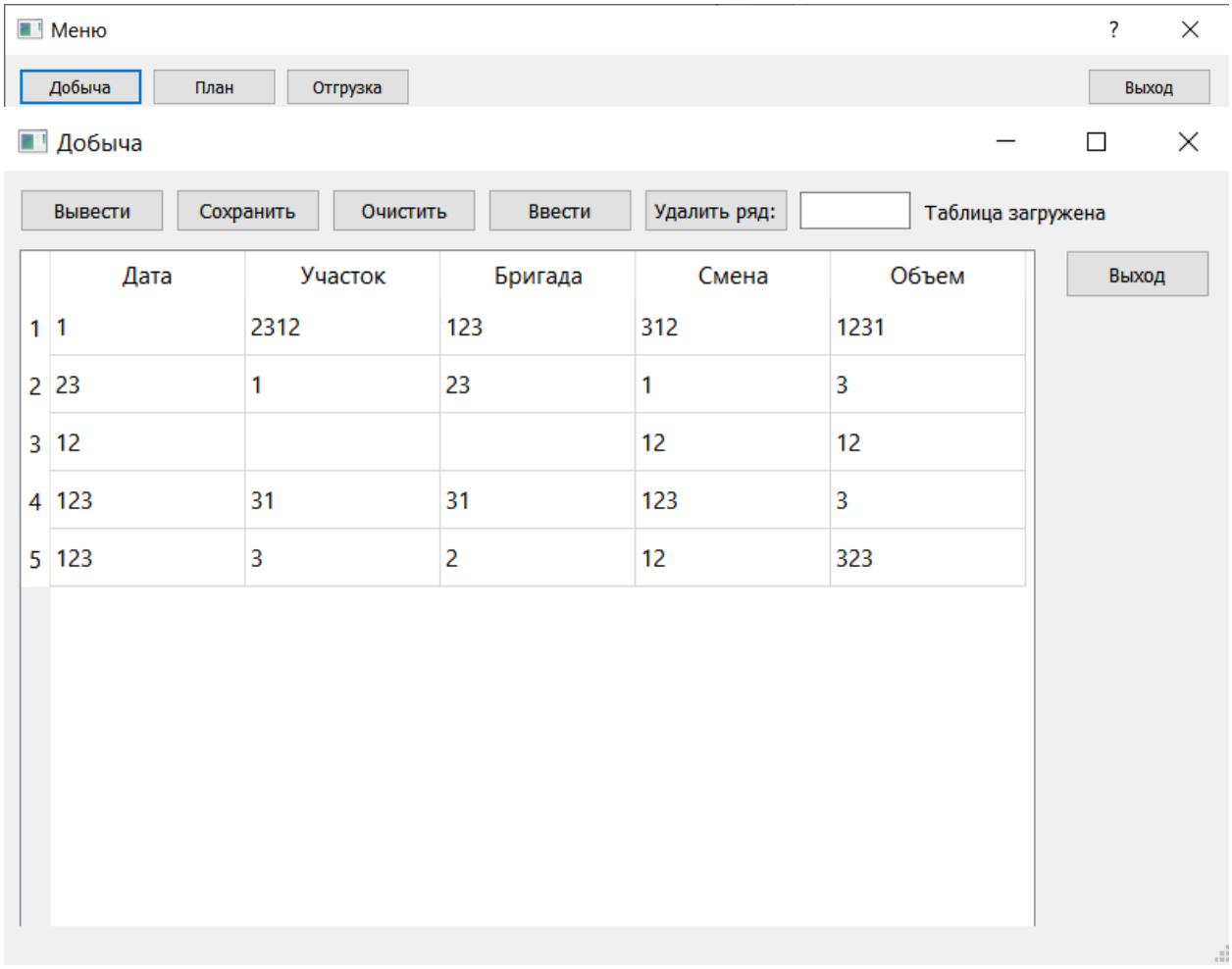
```
    <rect>
     <x>720</x>
     <y>10</y>
     <width>93</width>
     <height>28</height>
    </rect>
   </property>
   <property name="text">
    <string>Выход</string>
   </property>
  </widget>
 </widget>
 <resources/>
 <connections/>
</ui>
```

UML – диаграмма

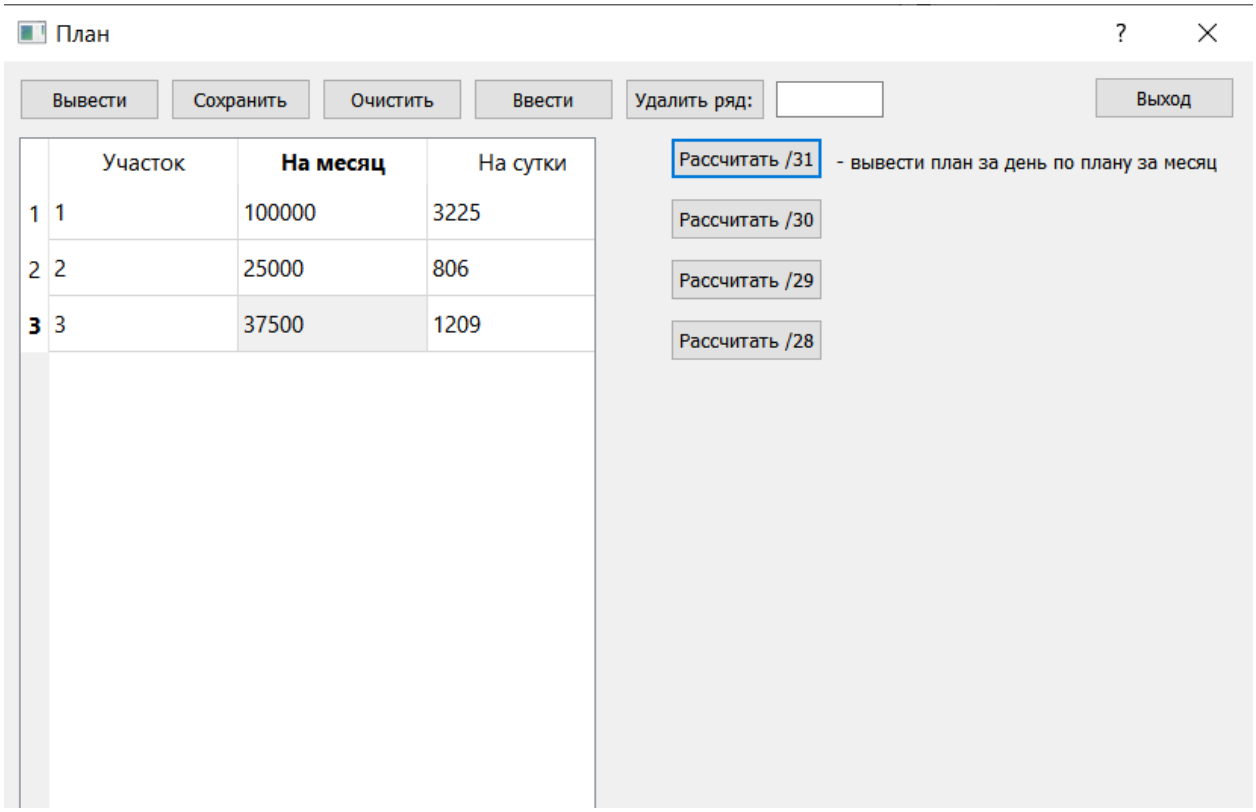


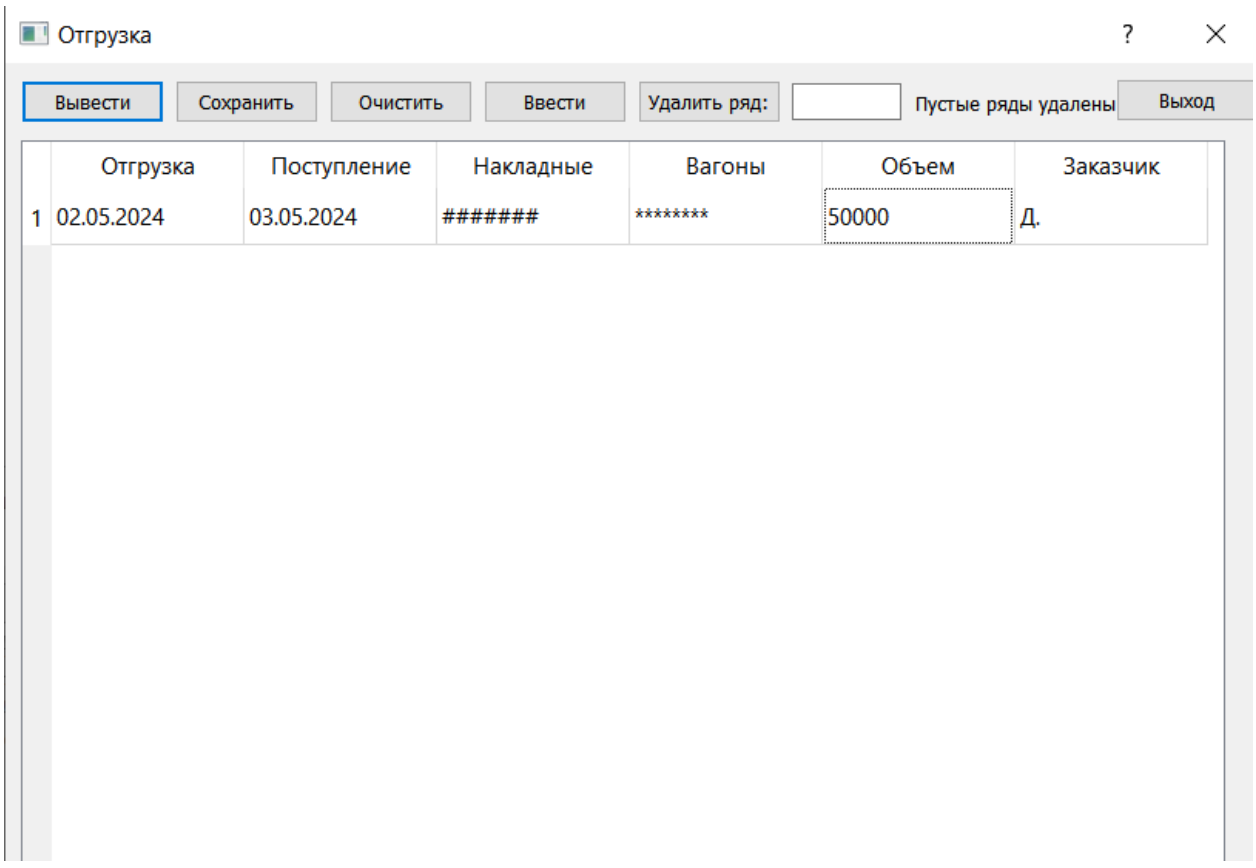
Внешний вид программы:





	Дата	Участок	Бригада	Смена	Объем
1	1	2312	123	312	1231
2	23	1	23	1	3
3	12			12	12
4	123	31	31	123	3
5	123	3	2	12	323





Видео: https://youtu.be/ycjcySVr6qs

Снято с использованием OBS Studio.