**Министерство образования и науки Российской Федерации**

**Пермское федеральное государственное бюджетное образовательное учреждение высшего образования «Пермский Национальный Исследовательский Политехнический Университет»**

**Электротехнический факультет**

**Кафедра «Информационные технологии и автоматизированные системы»**

**ОТЧЁТ**

По творческой работе на тему

«Разработка калькулятора и решение задачи о коммивояжёре»

Вариант №1

Выполнил студент группы РИС-20-1б

Азмагулов Артём Вадимович

Проверил доцент кафедры ИТАС

Полякова О.А.

Пермь 2021

**Цель работы** –продемонстрировать достижения, полученных студентом за время обучения на первом курсе.

**Постановка задачи**

Задача – реализовать на языке C++ с использованием технологии Qt два проекта, представляющих собой два приложения с графическим интерфейсом:

1. Узкоспециализированный калькулятор;
2. Система, представляющее решение задачи о коммивояжёре;

Исходные данные для **варианта №1**:

1. Специализация калькулятора: Работа с курсами валют:
   1. Изменение курса валют ;
   2. перевод между долларом, евро и рублем;
   3. отмена последней операции;
   4. сброс результата;
2. Специализация второй творческой работы: Нахождение кратчайшего пути полного графа:
   1. Отобразить введенный пользователем граф;
   2. Вывести длину маршрута коммивояжёра;
   3. Реализовать возможность редактирования графа:
      1. Добавление новых вершин;
      2. Изменение расстояний между вершинами;

**Анализ задачи**

**Часть №1: калькулятор курсов валют:**

**Валютный курс** — цена ([котировка](https://ru.wikipedia.org/wiki/%D0%9A%D0%BE%D1%82%D0%B8%D1%80%D0%BE%D0%B2%D0%BA%D0%B0)) денежной единицы одной страны, выраженная в денежной единице другой страны.

Валютный курс может определяться либо в результате взаимодействия рыночных сил, либо законодательно. В первом случае курс зависит от спроса на валюту и ее предложения на валютном рынке, во втором случае курс устанавливается государством.

Основываясь на данных фактах необходимо реализовать функции по приему константных значений доллара и евро в отношении рубля и объявить глобальные переменные по приему соответствующих данных. Однако учитывая факт того, что реализация калькулятора происходит за счет выбора нужного окна ввода путем нажатия кнопки RadioButton, то перед присвоением того или иного значения необходимо пройти проверку на нажатие соответствующих клавиш.

Соответственно, реализация функций по приему константных выражений выглядит следующим образом:

void MainWindow::**digits\_numbers**()

{

if(ui->radioButton\_Osnov->isChecked()

){

QPushButton \*button =( QPushButton \*)sender();

double all\_numbers;

QString new\_label;

if(((button->text())=="0")&&(ui->result\_show->text().contains((".")))){

ui->result\_show->setText(ui->result\_show->text()+button->text());

}

else{

all\_numbers=( ui->result\_show->text()+button->text()).toDouble();

impot=all\_numbers;

new\_label=QString::number(all\_numbers,'g',15);

ui->result\_show->setText(new\_label);}

}

if(ui->radioButtonDol->isChecked()

){

QPushButton \*button1 =( QPushButton \*)sender();if(((button1->text())=="0")&&(ui->labelDol->text().contains((".")))){

ui->labelDol->setText(ui->labelDol->text()+button1->text());

}

else{

double all\_numbers2;

QString new\_label2;

all\_numbers2=( ui->labelDol->text()+button1->text()).toDouble();

dol=all\_numbers2;//!!!!

new\_label2=QString::number(all\_numbers2,'g',15);

ui->labelDol->setText(new\_label2);

}}

if(ui->radioButton\_Euro->isChecked()

){

QPushButton \*button2 =( QPushButton \*)sender();

double all\_numbers3;

QString new\_label3;

if(((button2->text())=="0")&&(ui->label\_Euro->text().contains((".")))){

ui->label\_Euro->setText(ui->label\_Euro->text()+button2->text());

}

else{

all\_numbers3=( ui->label\_Euro->text()+button2->text()).toDouble();

euro=all\_numbers3;//!!!!

new\_label3=QString::number(all\_numbers3,'g',15);

ui->label\_Euro->setText(new\_label3);

}}

}

**Функция по нажатию кнопки(.) :**

При реализации функции по добавлению точки необходимо было учитывать соответствующее окно ввода данных для изменения соответствующих констант и добавление к ним десятичной. Основываясь на этом кнопка была реализована с добавлением в конец знака “.”.

Реализация функции выглядит следующим образом:

void MainWindow::**on\_pushButton\_Tochka\_clicked**()

{

if(ui->radioButton\_Osnov->isChecked()

){

if(!(ui->result\_show->text().contains(("."))))

ui->result\_show->setText(ui->result\_show->text()+".");}

if(ui->radioButtonDol->isChecked()

){

if(!(ui->labelDol->text().contains(("."))))

ui->labelDol->setText(ui->labelDol->text()+".");}

if(ui->radioButton\_Euro->isChecked()

){

if(!(ui->label\_Euro->text().contains((".")))){

// euro=( ui->label\_Euro->text()).toDouble();

ui->label\_Euro->setText(ui->label\_Euro->text()+".");}}

}

**Клавиши ввода чисел:**

Ввод цифр осуществляется с помощью следующей функции:

Реализация в коде на языке программирования C++ выглядит следующим образом:

MainWindow::**MainWindow**(QWidget \*parent)

: QMainWindow(*parent*)

, ui(new Ui::MainWindow)

{

ui->setupUi(this);

connect(ui->pushButton\_0,SIGNAL(clicked()),this,SLOT(digits\_numbers()));

connect(ui->pushButton1,SIGNAL(clicked()),this,SLOT(digits\_numbers()));

connect(ui->pushButton\_2,SIGNAL(clicked()),this,SLOT(digits\_numbers()));

connect(ui->pushButton\_3,SIGNAL(clicked()),this,SLOT(digits\_numbers()));

connect(ui->pushButton\_4,SIGNAL(clicked()),this,SLOT(digits\_numbers()));

connect(ui->pushButton\_5,SIGNAL(clicked()),this,SLOT(digits\_numbers()));

connect(ui->pushButton\_6,SIGNAL(clicked()),this,SLOT(digits\_numbers()));

connect(ui->pushButton\_7,SIGNAL(clicked()),this,SLOT(digits\_numbers()));

connect(ui->pushButton\_8,SIGNAL(clicked()),this,SLOT(digits\_numbers()));

connect(ui->pushButton\_9,SIGNAL(clicked()),this,SLOT(digits\_numbers()));

}

**Реализация функции по удалению строки:**

При реализации кнопки AC необходимо было учитывать положение соответствующей кнопки RadioButton, и в зависимости от нажатой пользователем кнопки присваивать соответствующей строке значение “0”.

Реализация метода в коде:

void MainWindow::**on\_pushButton\_AC\_clicked**()

{

if(ui->radioButton\_Osnov->isChecked()

){

QString new\_label6;

res=0;

new\_label6=QString::number(res,'g',15);

ui->result\_show->setText(new\_label6);

impot=0;}

if(ui->radioButtonDol->isChecked()

){

QString new\_label7;

dol=0;

new\_label7=QString::number(dol,'g',15);

ui->labelDol->setText(new\_label7);

}

if(ui->radioButton\_Euro->isChecked()

){

QString new\_label8;

euro=0;

new\_label8=QString::number(euro,'g',15);

ui->label\_Euro->setText(new\_label8);

}

}

**Функция, откатывающая последнее действие пользователя:**

При создании этой функции была использована хитрость, при которой программой запоминалось последнее значение, с которым работал пользователь. При нажатии на соответствующую кнопку программа возвращала на место главного окна сохраненное значение.

Реализация метода в коде:

void MainWindow::**on\_pushButton\_back\_clicked**()

{

QString new\_label12;

new\_label12=QString::number(impot2,'g',15);

ui->result\_show->setText(new\_label12);

impot=impot2;

}

**Реализация функции кнопки равно:**

При нажатии на соответствующую кнопку происходит прием всех написанных пользователем значений и подсчёт конечного результата с его дальнейшим выводом на главном окне.

Реализация метода в коде:

void MainWindow::**on\_pushButton\_ravno\_clicked**()

{

if(ui->radioButton\_dollar->isChecked()

){

now=impot\*dol;

}

if(ui->radioButton\_12Evro->isChecked()

){

euro=( ui->label\_Euro->text()).toDouble();

now=impot\*euro;

}

if(ui->radioButton\_10Rub->isChecked()

){

now=impot;

}

if(ui->radioButtonDollar->isChecked()

){

res=now/dol;

}

if(ui->radioButton\_Evro->isChecked()

){

res=now/euro;

}

if(ui->radioButton\_Rub->isChecked()

){

res=now;

}

QString new\_label5;

new\_label5=QString::number(res,'g',15);

ui->result\_show->setText(new\_label5);

impot2=impot;

impot=res;

}

**Реализация секретной кнопки:**

При реализации данной кнопки необходимо было подключить встроенный в QT класс <QMediaPlayer> для дальнейшей работы с аудиозаписями. При подключении данного класса следовала реализация самой кнопки, которая заключалась в сравнение введенного пользователя значения в главном окне с добавленными в функцию методами. Все подключения музыки были реализованы как путь к QT.

Реализация метода в коде:

void MainWindow::**on\_pushButton\_secret\_clicked**()

{

if(impot==1988){

player->stop();

player->setMedia(QUrl::fromLocalFile(QDir::toNativeSeparators(" D:\\qtRes\\qwerty\\1988.mp3")));

player->setVolume(50);

player->play();

//player->setPlaybackRate(QMediaPlayer:)

}

if(impot==1998){

player->stop();

player->setMedia(QUrl::fromLocalFile(QDir::toNativeSeparators(" D:\\qtRes\\qwerty\\1998.mp3")));

player->setVolume(50);

player->play();

//player->setPlaybackRate(QMediaPlayer:)

}

if(impot==1985){

player->stop();

player->setMedia(QUrl::fromLocalFile(QDir::toNativeSeparators("D:\\qtRes\\qwerty\\1985.mp3")));

player->setVolume(50);

player->play();

//player->setPlaybackRate(QMediaPlayer:)

}

if(impot==1990){

player->stop();

player->setMedia(QUrl::fromLocalFile(QDir::toNativeSeparators("D:\\qtRes\\qwerty\\1990.mp3")));

player->setVolume(50);

player->play();

//player->setPlaybackRate(QMediaPlayer:)

}

if(impot==1999){

player->stop();

player->setMedia(QUrl::fromLocalFile(QDir::toNativeSeparators("D:\\qtRes\\qwerty\\1999.mp3")));

player->setVolume(50);

player->play();

//player->setPlaybackRate(QMediaPlayer:)

}

if(impot==1992){

player->stop();

player->setMedia(QUrl::fromLocalFile(QDir::toNativeSeparators("D:\\qtRes\\qwerty\\1992.mp3")));

player->setVolume(50);

player->play();

//player->setPlaybackRate(QMediaPlayer:)

}

if(impot==1989){

player->stop();

player->setMedia(QUrl::fromLocalFile(QDir::toNativeSeparators("D:\\qtRes\\qwerty\\1989.mp3")));

player->setVolume(50);

player->play();

//player->setPlaybackRate(QMediaPlayer:)

}

if(impot==1986){

player->stop();

player->setMedia(QUrl::fromLocalFile(QDir::toNativeSeparators("D:\\qtRes\\qwerty\\1986.mp3")));

player->setVolume(50);

player->play();

//player->setPlaybackRate(QMediaPlayer:)

}

if(impot==1987){

player->stop();

player->setMedia(QUrl::fromLocalFile(QDir::toNativeSeparators("D:\\qtRes\\qwerty\\1987.mp3")));

player->setVolume(50);

player->play();

//player->setPlaybackRate(QMediaPlayer:)

}

if(impot==1991){

player->stop();

player->setMedia(QUrl::fromLocalFile(QDir::toNativeSeparators("D:\\qtRes\\qwerty\\1991.mp3")));

player->setVolume(50);

player->play();

//player->setPlaybackRate(QMediaPlayer:)

}

if(impot==1993){

player->stop();

player->setMedia(QUrl::fromLocalFile(QDir::toNativeSeparators("D:\\qtRes\\qwerty\\1993.mp3")));

player->setVolume(50);

player->play();

//player->setPlaybackRate(QMediaPlayer:)

}

if(impot==1994){

player->stop();

player->setMedia(QUrl::fromLocalFile(QDir::toNativeSeparators("D:\\qtRes\\qwerty\\1994.mp3")));

player->setVolume(50);

player->play();

//player->setPlaybackRate(QMediaPlayer:)

}

if(impot==1995){

player->stop();

player->setMedia(QUrl::fromLocalFile(QDir::toNativeSeparators("D:\\qtRes\\qwerty\\1995.mp3")));

player->setVolume(50);

player->play();

//player->setPlaybackRate(QMediaPlayer:)

}

if(impot==1996){

player->stop();

player->setMedia(QUrl::fromLocalFile(QDir::toNativeSeparators("D:\\qtRes\\qwerty\\1996.mp3")));

player->setVolume(50);

player->play();

//player->setPlaybackRate(QMediaPlayer:)

}

if(impot==1997){

player->stop();

player->setMedia(QUrl::fromLocalFile(QDir::toNativeSeparators("D:\\qtRes\\qwerty\\1997.mp3")));

player->setVolume(50);

player->play();

//player->setPlaybackRate(QMediaPlayer:)

}

if(impot==2001){

player->stop();

player->setMedia(QUrl::fromLocalFile(QDir::toNativeSeparators("D:\\qtRes\\qwerty\\2001.mp3")));

player->setVolume(50);

player->play();

//player->setPlaybackRate(QMediaPlayer:)

}

if(impot==2000){

player->stop();

player->setMedia(QUrl::fromLocalFile(QDir::toNativeSeparators("D:\\qtRes\\qwerty\\2000.mp3")));

player->setVolume(50);

player->play();

//player->setPlaybackRate(QMediaPlayer:)

}

if(impot==2002){

player->stop();

player->setMedia(QUrl::fromLocalFile(QDir::toNativeSeparators("D:\\qtRes\\qwerty\\2002.mp3")));

player->setVolume(50);

player->play();

//player->setPlaybackRate(QMediaPlayer:)

}

if(impot==2003){

player->stop();

player->setMedia(QUrl::fromLocalFile(QDir::toNativeSeparators("D:\\qtRes\\qwerty\\2003.mp3")));

player->setVolume(50);

player->play();

//player->setPlaybackRate(QMediaPlayer:)

}

if(impot==2004){

player->stop();

player->setMedia(QUrl::fromLocalFile(QDir::toNativeSeparators("D:\\qtRes\\qwerty\\2004.mp3")));

player->setVolume(50);

player->play();

//player->setPlaybackRate(QMediaPlayer:)

}

if(impot==2005){

player->stop();

player->setMedia(QUrl::fromLocalFile(QDir::toNativeSeparators("D:\\qtRes\\qwerty\\2005.mp3")));

player->setVolume(50);

player->play();

//player->setPlaybackRate(QMediaPlayer:)

}

if(impot==2006){

player->stop();

player->setMedia(QUrl::fromLocalFile(QDir::toNativeSeparators("D:\\qtRes\\qwerty\\2006.mp3")));

player->setVolume(50);

player->play();

//player->setPlaybackRate(QMediaPlayer:)

}

if(impot==2007){

player->stop();

player->setMedia(QUrl::fromLocalFile(QDir::toNativeSeparators("D:\\qtRes\\qwerty\\2007.mp3")));

player->setVolume(50);

player->play();

//player->setPlaybackRate(QMediaPlayer:)

}

if(impot==2008){

player->stop();

player->setMedia(QUrl::fromLocalFile(QDir::toNativeSeparators("D:\\qtRes\\qwerty\\2008.mp3")));

player->setVolume(50);

player->play();

//player->setPlaybackRate(QMediaPlayer:)

}

if(impot==2009){

player->stop();

player->setMedia(QUrl::fromLocalFile(QDir::toNativeSeparators("D:\\qtRes\\qwerty\\2009.mp3")));

player->setVolume(50);

player->play();

//player->setPlaybackRate(QMediaPlayer:)

}

if(impot==2010){

player->stop();

player->setMedia(QUrl::fromLocalFile(QDir::toNativeSeparators("D:\\qtRes\\qwerty\\2010.mp3")));

player->setVolume(50);

player->play();

//player->setPlaybackRate(QMediaPlayer:)

}

if(impot==0){

player->stop();

}

}

**Полная графическая составляющая программы:**

Интерфейс калькулятора реализован средствами технологии Qt и IDE Qt Creator, позволяющим верстать элементы графического интерфейса без необходимости писать код вручную.

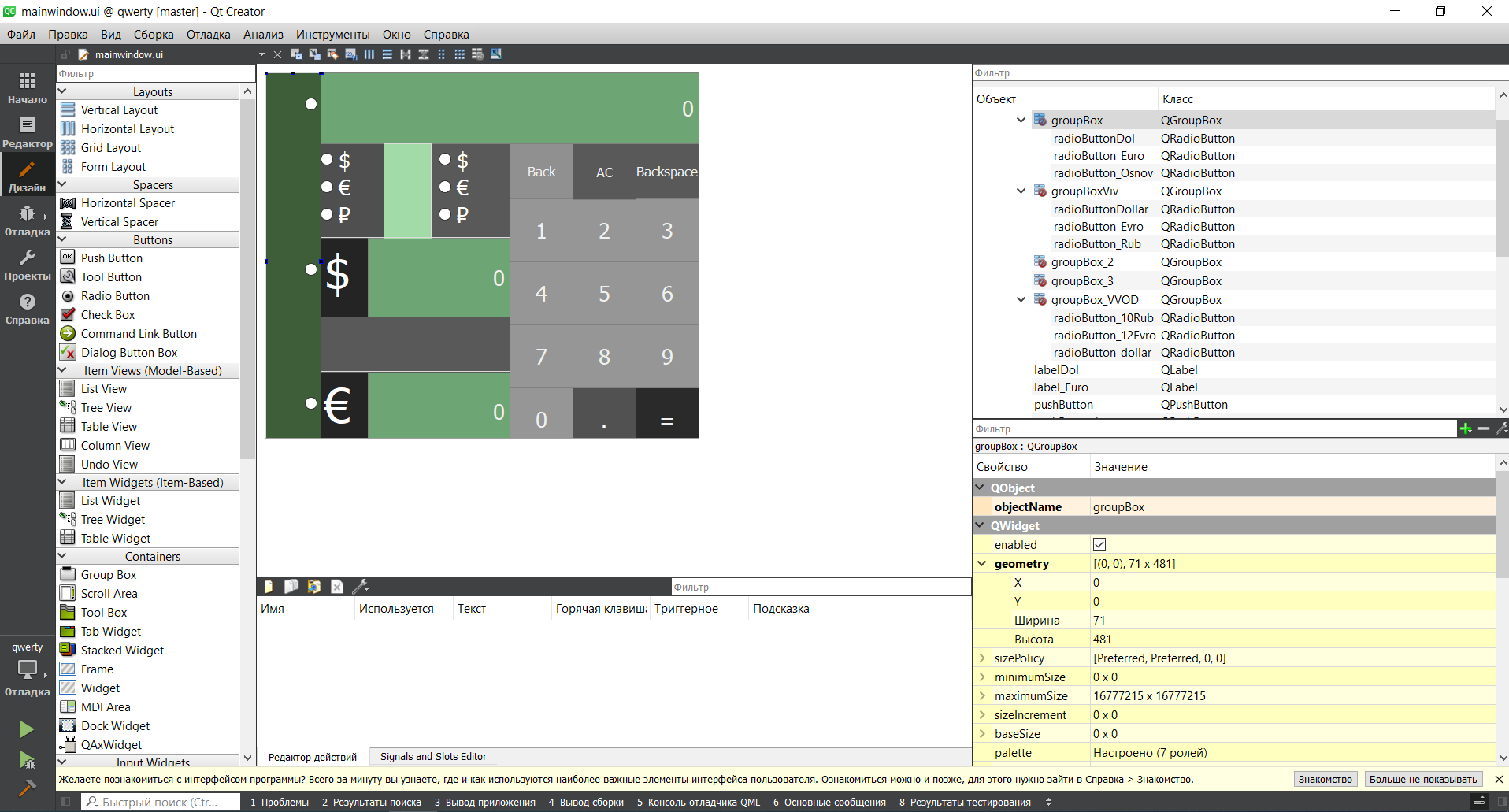


Рисунок - конструктор графического интерфейса в Qt Creator

Функционал интерфейса калькулятора:

* Ввод данных в приложения посредством нажатия клавиш с цифрами;
* Отображение вводимых данных в режиме реального времени;
* Вывод полученных данных после команд, реализованных в виде рычагов управления приложением, посредством нажатия кнопок;

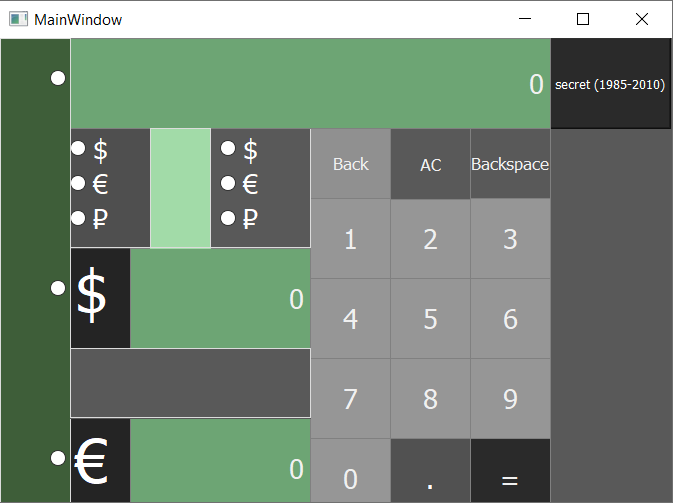
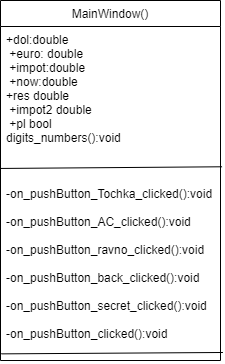


Рисунок - логическое разделение интерфейса калькулятора

UML-диаграмма приложения:



**Часть №2: решение задачи о коммивояжёре:**

**Граф** – это математическая абстракция некоторой реальной системы объектов любой природы, обладающих парными связями. Граф представляет собой математический объект как совокупность двух множеств, множества самих объектов, вершин, и множеством их парных связей, рёбер.

Город и дороги между ними, исходя из определения, данного выше, представляются в виде вершин и рёбер, соответственно. Ребро может иметь одно или два направления, также ребро может иметь свой вес (в данном случае, вес ребра означает расстояние между двумя городами).

**Представление графа в компьютере:**

Поскольку граф представляют в графическом виде, необходимо преобразовать графический представление в числовое, так как компьютер не понимает, что изображено на картинке. Одним из способов представления графов является представление при помощи матрицы смежности.

Матрица смежности – это таблица чисел, размером n\*n, где n – количество вершин в графе. Сами числа в таблице – это значения весов рёбер, соединяющих вершины графа. Каждая строка – номер начальной вершины ребра, каждый столбец – номер конечной вершины ребра. То есть, число, которое находится в 3-й строке и 4-м столбце, число равное длине ребра из вершины №3 в вершину №4.

Если связь отсутствует, то в ячейку записывается число 0. По вышеописанным правилам, выданный граф принимает следующий вид:

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
|  | 1 | 2 | 3 | 4 | 5 | 6 | 7 |
| 1 | 0 | 0 | 0 | **26** | **9** | 0 | 0 |
| 2 | 0 | 0 | 0 | **17** | 0 | 0 | **23** |
| 3 | 0 | 0 | 0 | 0 | **13** | **5** | 0 |
| 4 | **26** | **17** | 0 | 0 | 0 | **42** | 0 |
| 5 | **9** | 0 | **13** | 0 | 0 | 0 | **30** |
| 6 | 0 | 0 | **5** | **42** | 0 | 0 | **19** |
| 7 | 0 | **23** | 0 | 0 | **30** | **19** | 0 |

В языках программирования таблицы представляются в виде двумерных массивов или векторов. В моем же случае было решено выбрать реализацию через двумерный массив.

int mat[4][4] = { 1000,5,11,9,

10,1000,8,7,

7,14,1000,8,

12,6,15,1000 };

**Суть задачи о коммивояжёре:**

Постановка задачи о коммивояжёре гласит следующее: коммивояжёру, человеку, что приходит к разным людям и что-то предлагает, пересылает. У него стоит задача в котором ему необходимо:

1. Посетить все города ровно 1 раз;
2. Дойти до указанного пользователем города , кратчайшим путем;

Задача была решена методом перебора, с подсчетом всех возможных путей и их кратчайшего пути. Максимальное принимаемое количество графов-7.

int way[120];

long int var = 0;

long int min = 10000;

long int min\_way;

for(int i=0;i< ui->tableWidget\_2->rowCount();i++)

for(int j=0;j<ui->tableWidget\_2->columnCount();j++){

int h=(ui->tableWidget\_2->item(i,j)->text()).toInt();

if((h<=0)||(h>=1000)){

mat[i][j]=1000;

}

if((h>0)||(h<1000)){

mat[i][j]=h;

}

}

first--;

last--;

switch (col) {

case 3:

for (int i2 = 0; i2 < col; i2++) {

if ((first != i2) && (first != last) &&

(i2 != last)

) {

way[var] = ((first + 1) \* 10 + (i2 + 1)) \* 10 + last + 1;

//cout << way[var] << endl;

if (mat[first][i2] + mat[i2][last] < min) {

min = mat[first][i2] + mat[i2][last] ;

min\_way = way[var];

}

var++;

}

}

case 4:

for (int i2 = 0; i2 <col; i2++) {

for (int i3 = 0; i3 <col; i3++) {

if ((first != i2) && (first != i3) && (first != last) &&

(i2 != i3) && (i2 != last) &

(i3 != last)) {

way[var] = ((((first + 1) \* 10 + (i2 + 1)) \* 10 + (i3 + 1)) \* 10) + last + 1;

if (mat[first][i2] + mat[i2][i3] + mat[i3][last] < min) {

min = mat[first][i2] + mat[i2][i3] + mat[i3][last];

min\_way = way[var];

}

var++;

}

}

}

case 5:

for (int i2 = 0; i2 < col; i2++) {

for (int i3 = 0; i3 < col; i3++) {

for (int i4 = 0; i4 < col; i4++) {

if ((first != i2) && (first != i3) && (first != i4) && (first != last) &&

(i2 != i4) && (i2 != i3) && (i2 != last) &&

(i3 != i4) && (i3 != last) &&

(i4 != last)) {

way[var] = (((((first + 1) \* 10 + (i2 + 1)) \* 10 + (i3 + 1)) \* 10) + (i4 + 1)) \* 10 + last + 1;

//cout << way[var] << endl;

if (mat[first][i2] + mat[i2][i3] + mat[i3][i4] + mat[i4][last] < min) {

min = mat[first][i2] + mat[i2][i3] + mat[i3][i4] + mat[i4][last];

min\_way = way[var];

}

var++;

}

}

}

}

case 6:

for (int i2 = 0; i2 < col; i2++) {

for (int i3 = 0; i3 <col; i3++) {

for (int i4 = 0; i4 < col; i4++) {

for (int i5 = 0; i5 < col; i5++) {

if ((first != i2) && (first != i3) && (first != i4) && (first != i5) && (first != last) &&

(i2 != i4) && (i2 != i3) && (i2 != i5) && (i2 != last) &&

(i3 != i4) && (i3 != last) && (i3 != i5) &&

(i4 != last) && (i4 != i5)

&& (i5 != last)) {

way[var] = ((((((first + 1) \* 10 + (i2 + 1)) \* 10 + (i3 + 1)) \* 10) + (i4 + 1)) \* 10 + (i5 + 1)) \* 10 + last + 1;

//cout << way[var] << endl;

if (mat[first][i2] + mat[i2][i3] + mat[i3][i4] + mat[i4][i5] + mat[i5][last] < min) {

min = mat[first][i2] + mat[i2][i3] + mat[i3][i4] + mat[i4][i5] + mat[i5][last];

min\_way = way[var];

}

var++;

}

}

}

}

}

case 7:

for (int i2 = 0; i2 < col; i2++) {

for (int i3 = 0; i3 < col; i3++) {

for (int i4 = 0; i4 < col; i4++) {

for (int i5 = 0; i5 < col; i5++) {

for (int i6 = 0; i6 < col; i6++) {

if ((first != i2) && (first != i3) && (first != i4) && (first != i5) && (first != last) && (first != i6) &&

(i2 != i4) && (i2 != i3) && (i2 != i5) && (i2 != last) && (i2 != i6) &&

(i3 != i4) && (i3 != last) && (i3 != i5) && (i3 != i6) &&

(i4 != last) && (i4 != i5) && (i4 != i6)

&& (i5 != last) && (i5 != i6)

&& (i6 != last)) {

way[var] = (((((((first + 1) \* 10 + (i2 + 1)) \* 10 + (i3 + 1)) \* 10) + (i4 + 1)) \* 10 + (i5 + 1)) \* 10 + (i6 + 1)) \* 10 + last + 1;

//cout << way[var] << endl;

if (mat[first][i2] + mat[i2][i3] + mat[i3][i4] + mat[i4][i5] + mat[i5][i6] + mat[i6][last] < min) {

min = mat[first][i2] + mat[i2][i3] + mat[i3][i4] + mat[i4][i5] + mat[i6][last] + mat[i5][i6];

min\_way = way[var];

}

var++;

}

}

}

}

}

}

}

Таким образом, на выходе получится число, длина маршрута коммивояжёра и число – список вершин, через которые он проходит. Это и есть ответ на задачу о коммивояжёре.

Однако помимо числового ответа, необходимо реализовать графическое представление решения данной задачи. Для этого будут применены технологии OpenGL внутри Qt.

Поскольку граф может быть любым, с любым количеством вершин, с любыми связями, наиболее удобным и универсальным методом расположения вершин – это расположение вершин по окружности.

Для этого достаточно поделить окружность на количество частей, равному количеству вершин графа и расставить по полученных точкам вершины. Зная координаты этих вершин и их связи (обращаемся к матрице смежности) можно отрисовать рёбра соединяющие вершины.

void MainWindow::***paintEvent***(QPaintEvent\*)//Визуализация графа

{

if(ris==true){

QPainter painter;

painter.begin(this);

int len=col; //кол-во узлов в графе

int R=300; //радиус графа

int r=40; //радиус узла

double pi=3.1415;

int HeightText=30;

int xa=1000; //отступ от вертикали

int ya=600; //отступ от горизонтали

//ouble angle=pi/8;//угол раствора стрелки

//int ls=30; //длина кончика стрелки

int disp=60; //разброс подписей ребер

front=QBrush(QColor(127,255,212));

back=QBrush(QColor(200,200,200));

textPen=QPen(QColor(33,33,33));

//painter.fillRect(event->rect(),back);

painter.setBrush(front);

painter.setPen(textPen);

painter.setFont(glFont);

textPen.setWidth(4);

for (int i=0;i<col;i++)

{

double a=pi\*3/2+pi\*2/col\*i; //угол в полярных координатах (узел)

int xc=cos(a)\*R+xa; //декартовы координата узла

int yc=sin(a)\*R+ya;

int c1=xc-20; //координаты текста

int c2=yc-HeightText/2;

int c3=40;

int c4=HeightText;

int mat1[256][256];

QString text; text.setNum(i+1);

painter.drawEllipse(QRectF(xc-r,yc-r,2\*r,2\*r));

painter.drawText(QRect(c1,c2,c3,c4),Qt::AlignCenter,text);

for (int k=0;k<col;k++)

{

if(i!=k){

mat1[i][k]=mat[i][k];

if(mat1[i][k]!=mat1[k][i]){

//линия

double a2=pi\*3/2+pi\*2/col\*k;

double xc2=cos(a2)\*R+xa;

double yc2=sin(a2)\*R+ya;

double l=sqrt((xc2-xc)\*(xc2-xc)+(yc2-yc)\*(yc2-yc));

int xl1=(xc2-xc)/l\*r+xc;

int yl1=(yc2-yc)/l\*r+yc;

int xl2=xc2-(xc2-xc)/l\*r;

int yl2=yc2-(yc2-yc)/l\*r;

painter.drawLine(xl1,yl1,xl2,yl2);

//текст

c1=(xl2+xl1)/2-20+qrand()%disp-disp/2;

c2=(yl2+yl1)/2-HeightText/2+qrand()%disp-disp/2;

c3=40;

c4=HeightText;

text.setNum(mat[i][k]);

painter.drawText(QRect(c1,c2,c3,c4),Qt::AlignCenter,text);

}

}

}

}

}

}

UML-диаграмма приложения:

