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**

**«Теория алгоритмов и структуры данных»**

**на тему**

**«АРМ таможенника»**

Выполнил студент гр. ИВТ-23-1б

Бушмаков Кирилл Дмитриевич

Проверил:

Доцент каф. ИТАС

Яруллин Денис Владимирович

(оценка) (подпись)

(дата)

г. Пермь, 2024

**Цель и задачи работы**

Целью данной работы реализация задачи коммивояжера и

**UML диаграмма**

На рисунке 1 изображена диаграмма класса АРМ специалиста

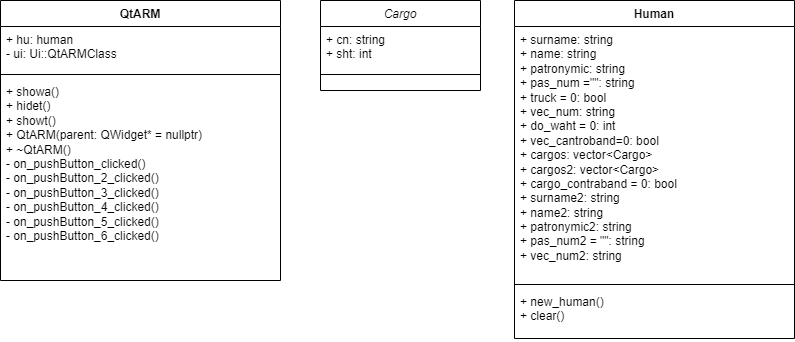


Рисунок 1

**Код программы для АРМ специалиста поливки полей**

**Файл QtArm.h**

#pragma once

#include <QtWidgets/QMainWindow>

#include "ui\_QtARM.h"

#include <vector>

using namespace std;

struct Cargo

{

string cn;

int sht;

};

class human

{

public:

string surname;

string name;

string patronymic;

string pas\_num ="";

bool truck = 0;

string vec\_num;

int do\_waht = 0;

bool vec\_cantroband=0;

vector<Cargo> cargos;

vector<Cargo> cargos2;

bool cargo\_contraband = 0;

string surname2;

string name2;

string patronymic2;

string pas\_num2 = "";

string vec\_num2;

void new\_human();

void clear();

};

class QtARM : public QMainWindow

{

Q\_OBJECT

public:

QtARM(QWidget \*parent = nullptr);

~QtARM();

human hu;

void showa();

void hidet();

void showt();

private:

Ui::QtARMClass ui;

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();

};

**Файл QtArm.cpp**

//♣↓

#include "QtARM.h"

#include <cstdlib>

#include <ctime>

using namespace std;

int dosmotr = 0;

int ver\_d = 0;

const char pass\_char[17] = { 'A','B' ,'C' ,'X' ,'P' ,'1' ,'2' ,'3' ,'4' ,'5' ,'6' ,'7','8','9','0','W','Q' };

const char num\_bukv[7] = { 'A' ,'B' ,'C' ,'D' ,'E' ,'W' ,'K' };

vector<string> cargo = { "Мешок", "Бочка","Ящик" };

bool c\_dos = 0, ca\_dos = 0;

string surnames[30] = {

"Иванов", "Петров", "Смирнов", "Соколов", "Кузнецов", "Попов", "Лебедев", "Козлов", "Новиков", "Морозов",

"Волков", "Зайцев", "Павлов", "Семенов", "Голубев", "Виноградов", "Богданов", "Воробьев", "Федоров", "Макаров",

"Андреев", "Козлов", "Ильин", "Степанов", "Сорокин", "Михайлов", "Калинин", "Герасимов", "Алексеев", "Лазарев"

};

string names[30] = {

"Иван", "Петр", "Александр", "Дмитрий", "Андрей",

"Сергей", "Михаил", "Юрий", "Николай", "Артем",

"Егор", "Илья", "Владимир", "Олег", "Григорий",

"Василий", "Степан", "Кирилл", "Тимофей", "Максим",

"Даниил", "Антон", "Павел", "Федор", "Виктор",

"Роман", "Глеб", "Ярослав", "Борислав", "Лев"

};

string patronymics[30] = {

"Александрович",

"Иванович",

"Дмитриевич",

"Сергеевич",

"Андреевич",

"Михайлович",

"Николаевич",

"Владимирович",

"Артемович",

"Алексеевич",

"Павлович",

"Егорович",

"Константинович",

"Григорьевич",

"Васильевич",

"Юрьевич",

"Олегович",

"Игоревич",

"Викторович",

"Романович",

"Денисович",

"Тимурович",

"Борисович",

"Евгеньевич",

"Леонидович",

"Станиславович",

"Валентинович",

"Федорович",

"Геннадьевич",

"Степанович"

};

QString co\_inf = "";

void human::new\_human()

{

srand(time(0));

int tp = abs(rand()) % 5;

int a = abs(rand()) % 30;

name = name2= names[a];

a = abs(rand()) % 30;

surname = surname2= surnames[a];

a = abs(rand()) % 30;

patronymic = patronymic2= patronymics[a];

vec\_num = vec\_num2= num\_bukv[abs(rand() % 7)] + to\_string(abs(rand() % 1000)) + num\_bukv[abs(rand() % 7)] + num\_bukv[abs(rand() % 7)];

for (int i = 0; i < 8; i++)

{

pas\_num += num\_bukv[abs(rand() % 7)];

}

pas\_num2 = pas\_num;

if (tp < 2)//truck

{

truck = true;

int cargo\_kol = 1 + abs(rand() % 3);

for (int i = 0; i < cargo\_kol; i++)

{

Cargo newCargo;

int cargo\_type = abs(rand() % 3);

// Проверка на уникальность груза

while (std::find\_if(cargos.begin(), cargos.end(), [&cargo\_type](const Cargo& c) { return c.cn == cargo[cargo\_type]; }) != cargos.end())

{

cargo\_type = abs(rand() % 3);

}

newCargo.cn = cargo[cargo\_type];

newCargo.sht = 1 + abs(rand() % 5);

cargos.push\_back(newCargo);

cargos2.push\_back(newCargo);

}

}

if (abs(rand() % 2) == 0)

{

int change\_type = abs(rand() % 6);

switch (change\_type)

{

case 0:

{

string new\_name;

do {

int a = abs(rand()) % 30;

new\_name = names[a];

} while (new\_name == name);

name2 = new\_name;

break;

}

case 1:

{

string new\_surname;

do {

int a = abs(rand()) % 30;

new\_surname = surnames[a];

} while (new\_surname == surname);

surname2 = new\_surname;

break;

}

case 2:

{

string new\_patronymic;

do {

int a = abs(rand()) % 30;

new\_patronymic = patronymics[a];

} while (new\_patronymic == patronymic);

patronymic2 = new\_patronymic;

break;

}

case 3:

{

string new\_num = vec\_num;

do {

new\_num = num\_bukv[abs(rand() % 7)] + to\_string(abs(rand() % 1000)) + num\_bukv[abs(rand() % 7)] + num\_bukv[abs(rand() % 7)];

} while (new\_num == vec\_num);

vec\_num2 = new\_num;

break;

}

case 4:

{

string new\_pas\_num = pas\_num;

do {

new\_pas\_num = "";

for (int i = 0; i < 8; i++)

{

new\_pas\_num += num\_bukv[abs(rand() % 7)];

}

} while (new\_pas\_num == pas\_num);

pas\_num2 = new\_pas\_num;

break;

}

case 5:

{

if (truck)

{

vector<Cargo> new\_cargos;

int cargo\_kol = 1 + abs(rand() % 3);

for (int i = 0; i < cargo\_kol; i++)

{

Cargo newCargo;

int cargo\_type = abs(rand() % 3);

// Проверка на уникальность груза

do {

cargo\_type = abs(rand() % 3);

} while (std::find\_if(new\_cargos.begin(), new\_cargos.end(), [&cargo\_type](const Cargo& c) { return c.cn == cargo[cargo\_type]; }) != new\_cargos.end());

newCargo.cn = cargo[cargo\_type];

newCargo.sht = 1 + abs(rand() % 5);

new\_cargos.push\_back(newCargo);

}

cargos2 = new\_cargos;

}

break;

}

default:

break;

}

do\_waht = 1;

}

if (abs(rand() % 5) == 0)

{

int ttt = (abs(rand() % 2));

if (ttt == 0)

{

vec\_cantroband = 1;

}

else if (ttt == 1 and truck)

{

cargo\_contraband = 1;

}

do\_waht = 2;

}

}

void human::clear()

{

surname.clear();

name.clear();

patronymic.clear();

pas\_num.clear();

truck = false;

vec\_num.clear();

do\_waht = 0;

vec\_cantroband = false;

cargos.clear();

cargos2.clear();

cargo\_contraband = false;

surname2.clear();

name2.clear();

patronymic2.clear();

pas\_num2.clear();

vec\_num2.clear();

}

QtARM::QtARM(QWidget \*parent)

: QMainWindow(parent)

{

ui.setupUi(this);

ui.go\_to->setStyleSheet("background-color: green;");

ui.arrest->setStyleSheet("background-color: red;");

hu.new\_human();

showa();

connect(ui.pushButton, &QPushButton::clicked, this, &QtARM::on\_pushButton\_clicked);

connect(ui.pushButton\_2, &QPushButton::clicked, this, &QtARM::on\_pushButton\_2\_clicked);

connect(ui.pushButton\_3, &QPushButton::clicked, this, &QtARM::on\_pushButton\_3\_clicked);

connect(ui.arrest, &QPushButton::clicked, this, &QtARM::on\_pushButton\_4\_clicked);

connect(ui.go\_to, &QPushButton::clicked, this, &QtARM::on\_pushButton\_5\_clicked);

connect(ui.razvorot, &QPushButton::clicked, this, &QtARM::on\_pushButton\_6\_clicked);

}

QtARM::~QtARM()

{

}

void QtARM::hidet()

{

ui.pushButton->hide();

ui.pushButton\_3->hide();

ui.label\_15->hide();

ui.label\_16->hide();

ui.textBrowser\_4->hide();

ui.textBrowser\_5->hide();

ui.textBrowser\_6->hide();

ui.textBrowser\_7->hide();

}

void QtARM::showt()

{

ui.pushButton->show();

ui.pushButton\_3->show();

ui.label\_15->show();

ui.label\_16->show();

ui.textBrowser\_4->show();

ui.textBrowser\_5->show();

ui.textBrowser\_6->show();

ui.textBrowser\_7->show();

}

void QtARM::showa()

{

QString inf = QString::fromLocal8Bit("Досмотров ") + QString::number(dosmotr) + QString::fromLocal8Bit("\nВерных досмотров ") + QString::number(ver\_d);

ui.textBrowser\_9->append(inf);

ui.passport\_surname->setText(QString::fromLocal8Bit(hu.surname.c\_str()));

ui.passport\_name->setText(QString::fromLocal8Bit(hu.name.c\_str()));

ui.passport\_patronymic->setText(QString::fromLocal8Bit(hu.patronymic.c\_str()));

ui.passport\_num->setText(QString::fromLocal8Bit(hu.pas\_num.c\_str()));

ui.passport\_surname\_2->setText(QString::fromLocal8Bit(hu.surname2.c\_str()));

ui.passport\_name\_2->setText(QString::fromLocal8Bit(hu.name2.c\_str()));

ui.passport\_patronymic\_2->setText(QString::fromLocal8Bit(hu.patronymic2.c\_str()));

ui.passport\_carnum\_2->setText(QString::fromLocal8Bit(hu.vec\_num.c\_str()));

ui.vechile\_num->setText(QString::fromLocal8Bit(hu.vec\_num2.c\_str()));

ui.passport\_num\_2->setText(QString::fromLocal8Bit(hu.pas\_num2.c\_str()));

if (hu.truck)

{

ui.textBrowser\_5->clear();

ui.car\_type->setText(QString::fromLocal8Bit("Грузовик"));

for (const Cargo& cargo : hu.cargos)

{

QString cargoInfo = QString::fromLocal8Bit(cargo.cn.c\_str()) + " " + QString::number(cargo.sht) + QString::fromLocal8Bit(" шт.\n");

ui.textBrowser\_5->append(cargoInfo);

}

showt();

}

else {

int ttk = rand() % 3;

string strh;

switch (ttk)

{

case(0):

strh = "Седан";

break;

case(1):

strh = "Внедорожник";

break;

case(2):

strh = "Пикап";

break;

default:

break;

}

ui.car\_type->setText(QString::fromLocal8Bit(strh.c\_str()));

hidet();

}

}

void QtARM::on\_pushButton\_clicked() {

for (const Cargo& cargose : hu.cargos2)

{

QString cargoInfos = QString::fromLocal8Bit(cargose.cn.c\_str()) + " " + QString::number(cargose.sht) + QString::fromLocal8Bit(" шт.\n");

ui.textBrowser\_6->append(cargoInfos);

}

}

void QtARM::on\_pushButton\_2\_clicked() {

if (hu.vec\_cantroband and !ca\_dos) {

ca\_dos = 1;

co\_inf +=QString::fromLocal8Bit("Найдена контрабанда в ТС\n");

}

else if(!ca\_dos) {

ca\_dos = 1;

co\_inf += QString::fromLocal8Bit("Контрабанда в ТС не обнаружена\n");

}

ui.textBrowser\_8->clear();

ui.textBrowser\_8->append(co\_inf);

}

void QtARM::on\_pushButton\_3\_clicked() {

if (hu.cargo\_contraband and !c\_dos) {

c\_dos = 1;

co\_inf += QString::fromLocal8Bit("Найдена контрабанда в грузе\n");

}

else if(!c\_dos) {

c\_dos = 1;

co\_inf += QString::fromLocal8Bit("Контрабанда в грузе не обнаружена\n");

}

ui.textBrowser\_8->clear();

ui.textBrowser\_8->append(co\_inf);

}

void QtARM::on\_pushButton\_4\_clicked() {

if (hu.do\_waht == 2) { ver\_d++; }

hu.clear();

hu.new\_human();

dosmotr++;

co\_inf = "";

ui.textBrowser\_9->clear();

ui.textBrowser\_8->clear();

ui.textBrowser\_5->clear();

ui.textBrowser\_6->clear();

c\_dos = ca\_dos = 0;

showa();

}

void QtARM::on\_pushButton\_5\_clicked() {

if (hu.do\_waht == 0) { ver\_d++; }

hu.clear();

hu.new\_human();

dosmotr++;

co\_inf = "";

ui.textBrowser\_9->clear();

ui.textBrowser\_8->clear();

ui.textBrowser\_5->clear();

ui.textBrowser\_6->clear();

c\_dos = ca\_dos = 0;

showa();

}

void QtARM::on\_pushButton\_6\_clicked() {

if (hu.do\_waht == 1) { ver\_d++; }

hu.clear();

hu.new\_human();

dosmotr++;

co\_inf = "";

ui.textBrowser\_9->clear();

ui.textBrowser\_8->clear();

ui.textBrowser\_5->clear();

ui.textBrowser\_6->clear();

c\_dos = ca\_dos = 0;

showa();

}

**Файл main.cpp**

#include "QtGraphs.h"

#include <QtWidgets/QApplication>

int main(int argc, char \*argv[])

{

QApplication a(argc, argv);

QtGraphs w;

w.show();

return a.exec();

}

**Демонстрация работы программ:**

https://youtu.be/M-STZWgzDb4