**ФЕДЕРАЛЬНОЕ АГЕНСТВО СВЯЗИ**

Ордена Трудового Красного Знамени

федеральное государственное бюджетное

образовательное учреждение высшего образования

«Московский Технический Университет Связи и Информатики»  
(МТУСИ)

**факультет ОТФ-2**

**кафедра информатики**

**КУРСОВАЯ РАБОТА**

По дисциплине: Информатика

Выполнила:

Студентка 1 курса

Группы БСТ-1602

Серкова Олеся

Вариант №13

Преподаватель,

принимающий работу:

Скрыпникова М. К.

Москва, 2017

Оглавление

[1.Общее задание на курсовую работу 3](#_Toc484646985)

[2.Индивидуальное задание. 3](#_Toc484646986)

[3.Пояснения к функциям. 4](#_Toc484646988)

[4.Код программы. 4](#_Toc484646989)

[4.1 Файл “Resourse.h” 4](#_Toc484646990)

[4.2 Файл “kursovayaSerkova.h” 5](#_Toc484646991)

[4.3 Файл “stdafx.h” 7](#_Toc484646992)

[4.4 Файл ”targetver.h” 7](#_Toc484646993)

[4.5 Файл “DrawPlane.cpp” 7](#_Toc484646994)

[4.6 Файл “InputText.cpp” 11](#_Toc484646995)

[4.7 Файл “kursovayaSerkova.cpp” 13](#_Toc484646996)

[4.8 Файл “Files.cpp” 23](#_Toc484646997)

[4.9 Файл “ObrabotkaInf.cpp” 26](#_Toc484646998)

[4.10 Файл “stdafx.cpp” 28](#_Toc484646999)

[5. Изображения работающей программы 28](#_Toc484647000)

# Общее задание на курсовую работу

Задание на проектирование программного приложения, включает в себя разработку для каждого индивидуального варианта статической графической рекламной заставки и функций накопления и сопровождения информации, заданной структуры, в файле на диске.

При проектировании приложения составить следующие функции:

Функцию, рисующую логотип (фирменный знак) заказчика. Функция должна рисовать изображение в относительных координатах и давать возможность через ее параметры менять размеры логотипа, его положение в активном окне и его цветовую гамму.

Функцию, делающую текстовую рекламную заставку с изображением логотипа заказчика (логотип рисуется разработанной вами функцией). Текст рекламной заставки, выполненный не менее 3 – 4 цветами.

Функции, обслуживающие работу с файлом, созданном в режиме .txt:

* запись информации в конец файла;
* просмотр всего файла;
* редактирование одного или нескольких (исходите из смысловой нагрузки задачи) полей информационной записи, найденной по указанному ключу;
* изменение содержания поля метки на удаление записи из файла ( по указанию пользователя найти требуемую запись и пометить ее на удаление или по указанию пользователя найти требуемую запись и снять метку на ее удаление);
* удаление записей из файла, помеченных в поле метки на удаление.

Функцию, решающую расчетную задачу. Эту функцию в свою очередь ТРЕБУЕТСЯ разбить на более простые функции, исходя из заданного алгоритма решения расчетной задачи и реализующие те или иные вычислительные процедуры расчетной задачи.

Информация для решения задачи должна накапливаться в файле по мере поступления.

В окно диалога About включить выходные данные о Вашей курсовой работе (вариант, исполнитель, руководитель) и Вашу фотографию.

# 2.Индивидуальное задание.

Фирма «Крылатые машины» заказала логотип и программу обработки информации. В файле накапливается следующая информация: бортовой номер самолета, марка самолета, место назначения, количество проданных билетов, стоимость одного билета. На основе этой информации необходимо ежедневно получать выходной документ в виде таблицы по самолетам одной марки. Информацию в таблицах разместить по убыванию суммарной выручки от продажи билетов.

# 3.Пояснения к функциям.

Функция void DrawPlane(HDC hDC, RECT rect, POINT beg, int palitra\_pic) рисует в окне логотип компании, основываясь на выбранной палитре и виде окна.

Функция void inputText (HWND hWnd, HDC hdc, RECT rect, int palitra) накладывает текст на окно.

Функция int strlen (TCHAR\* str) позволяет узнать длину строки.

Функция AirBeg GetAir (HWND hDlg) получает информацию о спортсменах из диалогового окна.

Функция void add(fstream &stream, AirBeg &airBeg, int n = 0) делает одиночную запись в файл.

Функция fstream& read(fstream&, AirBeg& airBeg, int = 0) читает из файла запись номер n.

Функция void SortedPrint(fstream& f,HWND hListBox) сравнивает суммарную стоимость для сортировки по убыванию.

Функция void MarkToDel(fstream& stream, AirBeg airBeg, int n) изменяет (добавляет или снимает) отметку на удаление выбранной строки.

Функция void DelMark(fstream& stream, AirBeg airBeg) удаляет отмеченные строки.

# 4.Код программы.

## 4.1 Файл “Resourse.h”

#define IDC\_MYICON 2

#define IDD\_KURSOVAYASERKOVA\_DIALOG 101

#define IDD\_ADD\_REC 102

#define IDS\_APP\_TITLE 103

#define IDD\_ABOUTBOX 103

#define IDM\_ABOUT 104

#define IDM\_EXIT 105

#define IDI\_KURSOVAYASERKOVA 107

#define IDI\_SMALL 108

#define IDC\_KURSOVAYASERKOVA 109

#define IDR\_MAINFRAME 128

#define IDD\_VIEW 130

#define IDD\_DIALOG1 131

#define IDD\_PUT\_WINNER 131

#define IDB\_BITMAP1 132

#define IDB\_I 132

#define IDB\_PHOTO 132

#define IDC\_GET\_NOM 1000

#define IDC\_GET\_MARKA 1001

#define IDC\_GET\_MESTO 1002

#define IDC\_GET\_BILET 1003

#define IDC\_GET\_PRICE 1004

#define IDC\_LIST1 1005

#define IDC\_REC\_ADD 1006

#define IDC\_WINNER 1007

#define IDC\_OTMENDEL 1008

#define IDC\_CHANGE 1009

#define IDC\_DELETE 1010

#define IDC\_POM 1011

#define ID\_32771 32771

#define ID\_32772 32772

#define ID\_32773 32773

#define ID\_FIRST\_PIC 32774

#define ID\_SECOND\_PIC 32775

#define ID\_TEXT\_GSY 32778

#define ID\_TEXT\_BDB 32780

#define ID\_TEXT\_TSH 32781

#define ID\_32782 32782

#define ID\_32783 32783

#define ID\_32784 32784

#define ID\_PIC\_BDB2 32785

#define ID\_PIC\_TSH2 32786

#define ID\_PIC\_GSY2 32787

#define VIEW 32788

#define IDC\_STATIC -1

// Next default values for new objects

//

#ifdef APSTUDIO\_INVOKED

#ifndef APSTUDIO\_READONLY\_SYMBOLS

#define \_APS\_NO\_MFC 1

#define \_APS\_NEXT\_RESOURCE\_VALUE 133

#define \_APS\_NEXT\_COMMAND\_VALUE 32789

#define \_APS\_NEXT\_CONTROL\_VALUE 1012

#define \_APS\_NEXT\_SYMED\_VALUE 110

#endif

#endif

## 4.2 Файл “kursovayaSerkova.h”

#pragma once

#include "resource.h"

#include <iostream>

#include <iomanip>

#include <fstream>

#include <time.h>

#include <iomanip>

using namespace std;

#define NF 30

#define NS 20

#define NL 5

#define NLINE (NF+NS+NL+1)

#define MAX 2\*NLINE

#define LenFam 20

#define LenSport 15

#define LenRec (LenFam+LenSport+LenFam+LenFam+LenFam+2+2)

#define Step 4

#define Filename "output.txt"

#define Temp "temp.txt"

#define fstream wfstream

struct AirBeg {

int nom;

TCHAR marka[NS];

TCHAR mesto[NF];

int bilet;

int price;

int del;

};

struct AirEnd {

int nom;

TCHAR marka[NS];

TCHAR mesto[NF];

int bilet;

int price;

int del;

};

struct StructEnd {

AirEnd airEnd;

TCHAR str[150];

};

void add(fstream &stream, AirBeg &airBeg, int n = 0);

fstream& read(fstream&, AirBeg& airBeg, int = 0);

void ChangeMarkToDelete(fstream& stream, HWND hDlg, HWND hListBox, int curItem);

void MarkToDel(fstream& stream, AirBeg airBeg, int n);

void DelMark(fstream& stream, AirBeg airBeg);

AirBeg GetAir(HWND hDlg);

void ReadAll(fstream& stream, HWND hListBox);

StructEnd obrabotkaInfo(AirBeg airBeg);

void putStudEndToStroka(StructEnd& u);

bool remove\_line(const char \*filename, size\_t index);

void inputText(HWND hWnd, HDC hdc, int palitra);

void DrawPlane(HDC hDC, RECT rect, POINT beg, int palitra\_pic);

AirBeg InputLineFromFile(int pos);

AirBeg OutputDataToFile(AirBeg airBeg);

void DelMark(fstream& stream);

void MarkToDel(fstream& stream, int n);

void changeStr(AirBeg& airBeg, fstream& stream, int n, TCHAR\* nCo, TCHAR\* nName\_1, TCHAR\* nName\_2, TCHAR\* nName\_3);

void SortedPrint(fstream& f,HWND hListBox);

## 4.3 Файл “stdafx.h”

// stdafx.h: включаемый файл для стандартных системных включаемых файлов

// или включаемых файлов для конкретного проекта, которые часто используются, но

// не часто изменяются

//

#pragma once

#include "targetver.h"

#define WIN32\_LEAN\_AND\_MEAN // Исключите редко используемые компоненты из заголовков Windows

// Файлы заголовков Windows:

#include <windows.h>

// Файлы заголовков C RunTime

#include <stdlib.h>

#include <malloc.h>

#include <memory.h>

#include <tchar.h>

// TODO: Установите здесь ссылки на дополнительные заголовки, требующиеся для программы

## 4.4 Файл ”targetver.h”

#pragma once

// Включение SDKDDKVer.h обеспечивает определение самой последней доступной платформы Windows.

// Если требуется выполнить сборку приложения для предыдущей версии Windows, включите WinSDKVer.h и

// задайте для макроса \_WIN32\_WINNT значение поддерживаемой платформы перед включением SDKDDKVer.h.

#include <SDKDDKVer.h>

## 4.5 Файл “DrawPlane.cpp”

#include "stdafx.h"

void DrawPlane(HDC hDC, RECT rect, POINT beg, int palitra\_pic)

{

//const int yCl=247;

//const int xCl=567;

const int yCl = (rect.bottom - rect.top);

//

const int xCl = (rect.right - rect.left);

//

const double dXPix = xCl / 16;

const double dYPix = yCl / 13;

SetMapMode(hDC, MM\_ISOTROPIC);

SetWindowExtEx(hDC, xCl, yCl, NULL);

SetViewportExtEx(hDC, xCl, -yCl, NULL);

SetViewportOrgEx(hDC, beg.x, yCl - beg.y, NULL);

COLORREF myCol[3][6] = { { RGB(0,0,255), RGB(0,0,0), RGB(0,191,255), RGB(30,144, 255), RGB(70,70,70), RGB(200,180,70) },

{ RGB(255, 20, 147), RGB(0,0,0), RGB(255, 99, 71), RGB(205, 92, 92), RGB(70,70,70), RGB(200,180,70) },

{ RGB(0, 128, 0), RGB(0,0,0), RGB(0, 255,127), RGB(60, 179, 113), RGB(70,70,70), RGB(200,180,70) }

};

//Создаем кисть

LOGBRUSH lb;

lb.lbColor = myCol[palitra\_pic][0];

lb.lbStyle = BS\_SOLID;

lb.lbHatch = 5;

HBRUSH hBrush = CreateBrushIndirect(&lb);

HBRUSH hOldBrush = (HBRUSH)SelectObject(hDC, hBrush);

//Турбина

POINT pl4[] = { { 2 \* dXPix, 4.75\*dYPix }, // 0

{ 2.25 \* dXPix, 7 \* dYPix }, // 1

{ 3.25 \* dXPix, 7 \* dYPix }, //2

{ 3.5 \* dXPix, 4.75\*dYPix },//3

{ 2 \* dXPix, 4.75 \* dYPix } }; //0

Polyline(hDC, pl4, sizeof(pl4) / sizeof(POINT));

FloodFill(hDC, 2.5 \* dXPix, 5 \* dYPix, myCol[palitra\_pic][1]);

//Турбина 2

POINT pl5[] = { { 12.5 \* dXPix, 4.75\*dYPix }, // 0

{ 12.75 \* dXPix, 7 \* dYPix }, // 1

{ 13.75 \* dXPix, 7 \* dYPix }, //2

{ 14 \* dXPix, 4.75\*dYPix },//3

{ 12.5 \* dXPix, 4.75 \* dYPix } }; //0

Polyline(hDC, pl5, sizeof(pl5) / sizeof(POINT));

FloodFill(hDC, 13 \* dXPix, 5 \* dYPix, myCol[palitra\_pic][1]);

//Меняем цвет кисти

lb.lbColor = myCol[palitra\_pic][2];

hBrush = CreateBrushIndirect(&lb);

DeleteObject(SelectObject(hDC, hBrush));

// Большие крылья

POINT pl[] = { { 8 \* dXPix, 3.5\*dYPix }, // 0

{ 0 , 7.5\*dYPix }, // 1

{ 16 \* dXPix, 7.5\*dYPix }, //2

{ 8 \* dXPix, 3.5\*dYPix } }; // 3

Polyline(hDC, pl, sizeof(pl) / sizeof(POINT));

HRGN kr = CreatePolygonRgn(pl, 4, ALTERNATE);

PaintRgn(hDC, kr);

DeleteObject(kr);

// Маленькие крылья

POINT pl2[] = { { 8 \* dXPix, 8 \* dYPix }, // 0

{ 6 \* dXPix , 10.5\*dYPix }, // 1

{ 8 \* dXPix, 10 \* dYPix }, //2

{ 10 \* dXPix, 10.5\*dYPix }, //3

{ 8 \* dXPix, 8 \* dYPix } }; // 4

Polyline(hDC, pl2, sizeof(pl2) / sizeof(POINT));

//Зарисовываем крылья

FloodFill(hDC, pl2[0].x , pl2[0].y + 1 \* dYPix, myCol[palitra\_pic][1]);

//Меняем цвет кисти

lb.lbColor = myCol[palitra\_pic][0];

hBrush = CreateBrushIndirect(&lb);

DeleteObject(SelectObject(hDC, hBrush));

//Корпус (большая часть)

POINT po2[] = { { 6 \* dXPix, 10 \* dYPix }, // 0

{ 10 \* dXPix, 0 }, // 1

{ 10 \* dXPix, 5 \* dYPix } }; // 2

int dCentr1X = (10 \* dXPix - 6 \* dXPix) / 2;

int dCentr1Y = (10 \* dYPix - 5 \* dYPix) / 2;

HRGN el = CreateEllipticRgn(6 \* dXPix, 10 \* dYPix, 10 \* dXPix, 0);

FillRgn(hDC, el, hBrush);

DeleteObject(el);

//Меняем цвет кисти

lb.lbColor = myCol[palitra\_pic][3];

hBrush = CreateBrushIndirect(&lb);

DeleteObject(SelectObject(hDC, hBrush));

//Корпус (маленькая часть)

POINT po3[] = { { 6.5 \* dXPix, 6 \* dYPix }, // 0

{ 9.5 \* dXPix, 0 }, // 1

{ 9.5 \* dXPix, 3 \* dYPix } }; // 2

int dCentr2X = (9.5 \* dXPix - 6.5 \* dXPix) / 2;

int dCentr2Y = (6 \* dYPix - 0) / 2;

for (int i = 0, dL = 14 \* dXPix; i < 1; i++)

{

Arc(hDC, po3[0].x + i \* dL, po3[0].y,

po3[1].x + i \* dL, po3[1].y,

po3[2].x + i \* dL, po3[2].y,

po3[2].x + i \* dL, po3[2].y);

}

FloodFill(hDC, 8 \* dXPix, 5 \* dYPix, myCol[palitra\_pic][1]);

FloodFill(hDC, 8 \* dXPix, 2 \* dYPix, myCol[palitra\_pic][1]);

//Меняем цвет кисти

lb.lbColor = myCol[palitra\_pic][0];

hBrush = CreateBrushIndirect(&lb);

DeleteObject(SelectObject(hDC, hBrush));

//Носик

POINT po4[] = { { 7.25 \* dXPix, 1.5 \* dYPix }, // 0

{ 8.75 \* dXPix, 0 }, // 1

{ 8.75 \* dXPix, 1.5 \* dYPix } }; // 2

int dCentr4X = (8.75 \* dXPix - 7.25 \* dXPix) / 2;

int dCentr4Y = (1.5 \* dYPix - 0) / 2;

for (int i = 0, dL = 14 \* dXPix; i < 1; i++)

{

Arc(hDC, po4[0].x + i \* dL, po4[0].y,

po4[1].x + i \* dL, po4[1].y,

po4[2].x + i \* dL, po4[2].y,

po4[2].x + i \* dL, po4[2].y);

}

FloodFill(hDC, 8 \* dXPix, 1 \* dYPix, myCol[palitra\_pic][1]);

//Меняем цвет кисти

lb.lbColor = myCol[palitra\_pic][5];

hBrush = CreateBrushIndirect(&lb);

DeleteObject(SelectObject(hDC, hBrush));

//Окно

POINT pl3[] = { { 8 \* dXPix, 2.5\*dYPix }, // 0

{ 7 \* dXPix, 3 \* dYPix }, // 1

{ 7 \* dXPix, 4 \* dYPix }, //2

{ 8 \* dXPix, 3.5\*dYPix }, //3

{ 9 \* dXPix, 4 \* dYPix }, //4

{ 9 \* dXPix, 3 \* dYPix }, //5

{ 8 \* dXPix, 2.5\*dYPix } }; //6

Polyline(hDC, pl3, sizeof(pl3) / sizeof(POINT));

FloodFill(hDC, pl3[0].x , pl3[0].y + 0.5 \* dYPix, myCol[palitra\_pic][1]);

//Меняем цвет кисти

lb.lbColor = myCol[palitra\_pic][4];

hBrush = CreateBrushIndirect(&lb);

DeleteObject(SelectObject(hDC, hBrush));

//Турбины (овальчики)

POINT po[] = { { 2 \* dXPix, 5.5 \* dYPix }, // 0

{ 3.5 \* dXPix, 4 \* dYPix }, // 1

{ 3.5 \* dXPix, 4.75\*dYPix } }; // 2

int dCentrX = (3.5 \* dXPix - 2 \* dXPix) / 2;

int dCentrY = (5.5 \* dYPix - 4 \* dYPix) / 2;

HRGN el1 = CreateEllipticRgn(2 \* dXPix, 5.5 \* dYPix, 3.5 \* dXPix, 4 \* dYPix);

FillRgn(hDC, el1, hBrush);

DeleteObject(el1);

POINT po0[] = { { 12.5 \* dXPix, 5.5 \* dYPix }, // 0

{ 14 \* dXPix, 4 \* dYPix }, // 1

{ 13.25 \* dXPix, 4.75\*dYPix } }; // 2

int dCentr0X = (14 \* dXPix - 12.5 \* dXPix) / 2;

int dCentr0Y = (5.5 \* dYPix - 4 \* dYPix) / 2;

HRGN el0 = CreateEllipticRgn(12.5 \* dXPix, 5.5 \* dYPix, 14 \* dXPix, 4 \* dYPix);

FillRgn(hDC, el0, hBrush);

DeleteObject(el0);

}

## 4.6 Файл “InputText.cpp”

#include "stdafx.h"

int strlen(TCHAR\* str)

{

int len;

for (len = 0; \*str; len++, str++);

return len;

}

void inputText(HWND hWnd, HDC hdc, int palitra)

{

PAINTSTRUCT ps;

TCHAR\* line[] = { TEXT("Крылатые машины."),

TEXT("Серкова Олеся"),

TEXT("8(985)891-63-01"),

TEXT("Небольшая частная авиакомпания.")

};

// макрос RGB задает палитру цветов от RGB(255,255,255)белый,

// до RGB(0,0,0) черный

COLORREF masColor[3][3] = { { RGB(0, 0, 255), RGB(30, 144, 255), RGB(25, 25, 112) },

{ RGB(255, 99, 71), RGB(250, 128, 114), RGB(255, 105, 180) },

{ RGB(0, 128, 0), RGB(0, 255, 127), RGB(225, 225, 0) }

};

HPEN hPen0 = CreatePen(PS\_SOLID, 6, masColor[palitra][0]);

HPEN hOldPen = (HPEN)SelectObject(hdc, hPen0);

static LOGFONT lf;

lf.lfPitchAndFamily = FIXED\_PITCH | FF\_MODERN;

// Шаг шрифта DEFAULT\_PITCH по умолчанию

lf.lfItalic = TRUE;

// запрашивается курсивный шрифт, если TRUE

lf.lfWeight = FW\_BOLD;

// жирность шрифта 0(FW\_DONTCARE),100,200,... 700,..,900

// значение 0 предписывает выбрать вес по умолчанию

lf.lfHeight = 70;

lf.lfCharSet = RUSSIAN\_CHARSET;

// код используемого набора символов

HFONT hFont0 = CreateFontIndirect(&lf);

HFONT hOldFont = (HFONT)SelectObject(hdc, hFont0);

SetTextColor(hdc, masColor[palitra][1]);

int x = 50, y = 30, dX = 80, dY = 80;

TextOut(hdc, x, y, line[0], strlen(line[0]));

POINT tabStop[] = { { x + dX,y + 2 \* dY },

{ x + dX,y + 3.5\*dY },

{ x + dX,y + 5 \* dY }

};

int fontHeight[] = { 40,60,60 };

MoveToEx(hdc, x, tabStop[0].y - 83, NULL);

LineTo(hdc, x + 7.7\*dX, tabStop[0].y - 83);

DeleteObject(SelectObject(hdc, hFont0));

//

for (int i = 0; i<3; ++i)

{

lf.lfPitchAndFamily = FIXED\_PITCH | FF\_MODERN;

// Шаг шрифта DEFAULT\_PITCH по умолчанию

lf.lfItalic = FALSE;

// запрашивается курсивный шрифт, если TRUE

lf.lfWeight = FW\_BOLD;

// жирность шрифта 0(FW\_DONTCARE),100,200,... 700,..,900

// значение 0 предписывает выбрать вес по умолчанию

lf.lfHeight = fontHeight[i];

lf.lfCharSet = RUSSIAN\_CHARSET;

// код используемого набора символов

hFont0 = CreateFontIndirect(&lf);

SelectObject(hdc, hFont0);

SetTextColor(hdc, masColor[palitra][i]);

TextOut(hdc, tabStop[i].x, tabStop[i].y, line[i + 1], strlen(line[i + 1]));

DeleteObject(SelectObject(hdc, hFont0));

}

DeleteObject(SelectObject(hdc, hOldPen));

DeleteObject(SelectObject(hdc, hOldFont));

}

## 4.7 Файл “kursovayaSerkova.cpp”

// kursovayaSerkova.cpp: определяет точку входа для приложения.

//

#include "stdafx.h"

#include <Windows.h>

#include "kursovayaSerkova.h"

#define MAX\_LOADSTRING 100

// Глобальные переменные:

HINSTANCE hInst; // текущий экземпляр

TCHAR szTitle[MAX\_LOADSTRING]; // Текст строки заголовка

TCHAR szWindowClass[MAX\_LOADSTRING]; // имя класса главного окна

AirBeg airBeg = { 0,TEXT(""),TEXT(""),0,0 };

// Отправить объявления функций, включенных в этот модуль кода:

ATOM MyRegisterClass(HINSTANCE hInstance);

BOOL InitInstance(HINSTANCE, int);

LRESULT CALLBACK WndProc(HWND, UINT, WPARAM, LPARAM);

INT\_PTR CALLBACK About(HWND, UINT, WPARAM, LPARAM);

INT\_PTR CALLBACK GetDataDlgProc(HWND , UINT , WPARAM , LPARAM );

INT\_PTR CALLBACK ViewDlgProc(HWND , UINT , WPARAM , LPARAM );

INT\_PTR CALLBACK PutDataDlgProc(HWND , UINT , WPARAM , LPARAM );

BOOL CALLBACK ChangeDataDlgProc(HWND , UINT , WPARAM , LPARAM );

int APIENTRY \_tWinMain(HINSTANCE hInstance,

HINSTANCE hPrevInstance,

LPTSTR lpCmdLine,

int nCmdShow)

{

UNREFERENCED\_PARAMETER(hPrevInstance);

UNREFERENCED\_PARAMETER(lpCmdLine);

// TODO: разместите код здесь.

MSG msg;

HACCEL hAccelTable;

// Инициализация глобальных строк

LoadString(hInstance, IDS\_APP\_TITLE, szTitle, MAX\_LOADSTRING);

LoadString(hInstance, IDC\_KURSOVAYASERKOVA, szWindowClass, MAX\_LOADSTRING);

MyRegisterClass(hInstance);

// Выполнить инициализацию приложения:

if (!InitInstance(hInstance, nCmdShow))

{

return FALSE;

}

hAccelTable = LoadAccelerators(hInstance, MAKEINTRESOURCE(IDC\_KURSOVAYASERKOVA));

// Цикл основного сообщения:

while (GetMessage(&msg, NULL, 0, 0))

{

if (!TranslateAccelerator(msg.hwnd, hAccelTable, &msg))

{

TranslateMessage(&msg);

DispatchMessage(&msg);

}

}

return (int)msg.wParam;

}

//

// ФУНКЦИЯ: MyRegisterClass()

//

// НАЗНАЧЕНИЕ: регистрирует класс окна.

//

// КОММЕНТАРИИ:

//

// Эта функция и ее использование необходимы только в случае, если нужно, чтобы данный код

// был совместим с системами Win32, не имеющими функции RegisterClassEx'

// которая была добавлена в Windows 95. Вызов этой функции важен для того,

// чтобы приложение получило "качественные" мелкие значки и установило связь

// с ними.

//

ATOM MyRegisterClass(HINSTANCE hInstance)

{

WNDCLASSEX wcex;

wcex.cbSize = sizeof(WNDCLASSEX);

wcex.style = CS\_HREDRAW | CS\_VREDRAW;

wcex.lpfnWndProc = WndProc;

wcex.cbClsExtra = 0;

wcex.cbWndExtra = 0;

wcex.hInstance = hInstance;

wcex.hIcon = LoadIcon(hInstance, MAKEINTRESOURCE(IDI\_KURSOVAYASERKOVA));

wcex.hCursor = LoadCursor(NULL, IDC\_ARROW);

wcex.hbrBackground = (HBRUSH)(COLOR\_WINDOW + 1);

wcex.lpszMenuName = MAKEINTRESOURCE(IDC\_KURSOVAYASERKOVA);

wcex.lpszClassName = szWindowClass;

wcex.hIconSm = LoadIcon(wcex.hInstance, MAKEINTRESOURCE(IDI\_SMALL));

return RegisterClassEx(&wcex);

}

//

// ФУНКЦИЯ: InitInstance(HINSTANCE, int)

//

// НАЗНАЧЕНИЕ: сохраняет обработку экземпляра и создает главное окно.

//

// КОММЕНТАРИИ:

//

// В данной функции дескриптор экземпляра сохраняется в глобальной переменной, а также

// создается и выводится на экран главное окно программы.

//

BOOL InitInstance(HINSTANCE hInstance, int nCmdShow)

{

HWND hWnd;

hInst = hInstance; // Сохранить дескриптор экземпляра в глобальной переменной

hWnd = CreateWindow(szWindowClass, szTitle, WS\_OVERLAPPEDWINDOW,

CW\_USEDEFAULT, 0, CW\_USEDEFAULT, 0, NULL, NULL, hInstance, NULL);

if (!hWnd)

{

return FALSE;

}

ShowWindow(hWnd, nCmdShow);

UpdateWindow(hWnd);

return TRUE;

}

//

// ФУНКЦИЯ: WndProc(HWND, UINT, WPARAM, LPARAM)

//

// НАЗНАЧЕНИЕ: обрабатывает сообщения в главном окне.

//

// WM\_COMMAND - обработка меню приложения

// WM\_PAINT -Закрасить главное окно

// WM\_DESTROY - ввести сообщение о выходе и вернуться.

//

//

LRESULT CALLBACK WndProc(HWND hWnd, UINT message, WPARAM wParam, LPARAM lParam)

{

int wmId, wmEvent;

PAINTSTRUCT ps;

static int picture = 1;

HDC hdc;

static int palitra\_text = 0, palitra\_pic = 0;

static HMENU hMenu = GetMenu(hWnd);

SetConsoleCP(1251);

SetConsoleOutputCP(1251);

switch (message)

{

case WM\_CREATE:

CheckMenuRadioItem(GetSubMenu(hMenu, 1), ID\_TEXT\_BDB,

ID\_TEXT\_GSY, ID\_TEXT\_BDB, MF\_BYCOMMAND);

CheckMenuRadioItem(GetSubMenu(hMenu, 1), ID\_PIC\_BDB2,

ID\_PIC\_GSY2, ID\_PIC\_BDB2, MF\_BYCOMMAND);

CheckMenuRadioItem(GetSubMenu(hMenu, 2), ID\_FIRST\_PIC,

ID\_SECOND\_PIC, ID\_FIRST\_PIC, MF\_BYCOMMAND);

EnableMenuItem(hMenu, ID\_TEXT\_BDB, MF\_BYCOMMAND | MFS\_GRAYED);

EnableMenuItem(hMenu, ID\_TEXT\_GSY, MF\_BYCOMMAND | MFS\_GRAYED);

EnableMenuItem(hMenu, ID\_TEXT\_TSH, MF\_BYCOMMAND | MFS\_GRAYED);

DrawMenuBar(hWnd);

break;

case WM\_COMMAND:

wmId = LOWORD(wParam);

wmEvent = HIWORD(wParam);

// Разобрать выбор в меню:

switch (wmId)

{

case VIEW:

DialogBox(hInst, MAKEINTRESOURCE(IDD\_VIEW), hWnd, ViewDlgProc);

break;

case ID\_FIRST\_PIC:

picture = 1;

CheckMenuRadioItem(GetSubMenu(hMenu, 2), ID\_FIRST\_PIC,

ID\_SECOND\_PIC, LOWORD(wParam), MF\_BYCOMMAND);

EnableMenuItem(hMenu, ID\_TEXT\_BDB, MF\_BYCOMMAND | MFS\_GRAYED);

EnableMenuItem(hMenu, ID\_TEXT\_GSY, MF\_BYCOMMAND | MFS\_GRAYED);

EnableMenuItem(hMenu, ID\_TEXT\_TSH, MF\_BYCOMMAND | MFS\_GRAYED);

//DrawMenuBar(hWnd);

break;

case ID\_SECOND\_PIC:

picture = 2;

CheckMenuRadioItem(GetSubMenu(hMenu, 2), ID\_FIRST\_PIC,

ID\_SECOND\_PIC, LOWORD(wParam), MF\_BYCOMMAND);

EnableMenuItem(hMenu, ID\_TEXT\_BDB, MF\_BYCOMMAND | MFS\_ENABLED);

EnableMenuItem(hMenu, ID\_TEXT\_GSY, MF\_BYCOMMAND | MFS\_ENABLED);

EnableMenuItem(hMenu, ID\_TEXT\_TSH, MF\_BYCOMMAND | MFS\_ENABLED);

//DrawMenuBar(hWnd);

break;

case ID\_PIC\_BDB2:

palitra\_pic = 0;

CheckMenuRadioItem(GetSubMenu(hMenu, 1), ID\_PIC\_BDB2,

ID\_PIC\_GSY2, LOWORD(wParam), MF\_BYCOMMAND);

break;

case ID\_PIC\_TSH2:

palitra\_pic = 1;

CheckMenuRadioItem(GetSubMenu(hMenu, 1), ID\_PIC\_BDB2,

ID\_PIC\_GSY2, LOWORD(wParam), MF\_BYCOMMAND);

break;

case ID\_PIC\_GSY2:

palitra\_pic = 2;

CheckMenuRadioItem(GetSubMenu(hMenu, 1), ID\_PIC\_BDB2,

ID\_PIC\_GSY2, LOWORD(wParam), MF\_BYCOMMAND);

break;

case ID\_TEXT\_BDB:

palitra\_text = 0;

CheckMenuRadioItem(GetSubMenu(hMenu, 1), ID\_TEXT\_BDB,

ID\_TEXT\_GSY, LOWORD(wParam), MF\_BYCOMMAND);

break;

case ID\_TEXT\_TSH:

palitra\_text = 1;

CheckMenuRadioItem(GetSubMenu(hMenu, 1), ID\_TEXT\_BDB,

ID\_TEXT\_GSY, LOWORD(wParam), MF\_BYCOMMAND);

break;

case ID\_TEXT\_GSY:

palitra\_text = 2;

CheckMenuRadioItem(GetSubMenu(hMenu, 1), ID\_TEXT\_BDB,

ID\_TEXT\_GSY, LOWORD(wParam), MF\_BYCOMMAND);

break;

case IDM\_ABOUT:

DialogBox(hInst, MAKEINTRESOURCE(IDD\_ABOUTBOX), hWnd, About);

break;

case IDM\_EXIT:

DestroyWindow(hWnd);

break;

default:

return DefWindowProc(hWnd, message, wParam, lParam);

}

InvalidateRect(hWnd, NULL, TRUE);

break;

case WM\_PAINT:

{

POINT beg = { 0, 0 };

hdc = BeginPaint(hWnd, &ps);

RECT rect;

switch (picture) {

case 1:

beg.x = 0; beg.y = 0;

GetClientRect(hWnd, &rect);

break;

case 2:

rect.top = 300;

rect.right = 767;

rect.bottom = 547;

rect.left = 200;

beg.x = 567; beg.y = -80;

inputText(hWnd, hdc, palitra\_text);

break;

}

DrawPlane(hdc, rect, beg, palitra\_pic);

EndPaint(hWnd, &ps);

break;

}

case WM\_DESTROY:

PostQuitMessage(0);

break;

default:

return DefWindowProc(hWnd, message, wParam, lParam);

}

return 0;

}

// Обработчик сообщений для окна "О программе".

INT\_PTR CALLBACK About(HWND hDlg, UINT message, WPARAM wParam, LPARAM lParam)

{

UNREFERENCED\_PARAMETER(lParam);

switch (message)

{

case WM\_INITDIALOG:

return (INT\_PTR)TRUE;

case WM\_COMMAND:

if (LOWORD(wParam) == IDOK || LOWORD(wParam) == IDCANCEL)

{

EndDialog(hDlg, LOWORD(wParam));

return (INT\_PTR)TRUE;

}

break;

}

return (INT\_PTR)FALSE;

}

INT\_PTR CALLBACK ViewDlgProc(HWND hDlg, UINT message, WPARAM wParam, LPARAM lParam)

{

UNREFERENCED\_PARAMETER(lParam);

static int nU = 0;

static HWND hListBox;

static LOGFONT lf;

HFONT hFont1;

static int ip;

int polow = 0;

int curItem=0;

cin.imbue(locale(".866"));

fstream stream(Filename, ios::in|ios::out);

if (!stream)

{

stream.close();

stream.open(Filename, ios::out);

stream.close();

stream.open(Filename, ios::in|ios::out);

}

stream.imbue(locale(".1251"));

switch (message)

{

case WM\_INITDIALOG:

hListBox = GetDlgItem(hDlg, IDC\_LIST1);

ReadAll(stream, hListBox);

SetFocus(hDlg);

lf.lfHeight = 16;

lstrcpy((LPWSTR)&lf.lfFaceName, TEXT("Courier"));

hFont1 = CreateFontIndirect(&lf);

SendMessage(hListBox, WM\_SETFONT, (WPARAM)hFont1, TRUE);

return (INT\_PTR)TRUE;

case WM\_COMMAND:

switch (LOWORD(wParam))

{

case WM\_INITDIALOG:

hListBox = GetDlgItem(hDlg, IDC\_LIST1);

break;

case IDOK:

nU = SendMessage(hListBox, LB\_GETCOUNT, 0, 0);

stream.close();

EndDialog(hDlg, LOWORD(wParam));

return (INT\_PTR)TRUE;

break;

case IDCANCEL:

nU = SendMessage(hListBox, LB\_GETCOUNT, 0, 0);

stream.close();

EndDialog(hDlg, LOWORD(wParam));

return (INT\_PTR)TRUE;

break;

case IDC\_WINNER:

DialogBox((HINSTANCE)GetModuleHandle(NULL), MAKEINTRESOURCE(IDD\_PUT\_WINNER), hDlg, PutDataDlgProc);

break;

case IDC\_CHANGE:

{

curItem = SendMessage(hListBox, LB\_GETCURSEL, 0, 0);

if (curItem != LB\_ERR)

{

DialogBox(hInst, MAKEINTRESOURCE(IDD\_ADD\_REC), hDlg, GetDataDlgProc);

airBeg.del = 0;

add(stream, airBeg, curItem + 1);

SendMessage(hListBox, LB\_RESETCONTENT, 0, 0);

ReadAll(stream, hListBox);

}

else

{

MessageBox(hDlg, L"сначала надо выделить элемент списка", L"ОШИБКА", MB\_OK);

break;

}

}

break;

/\*DialogBox((HINSTANCE)GetModuleHandle(NULL), MAKEINTRESOURCE(IDD\_CHANGE), hDlg, ChangeDataDlgProc);

break;\*/

case IDC\_OTMENDEL:

{

curItem = SendMessage(hListBox, LB\_GETCURSEL, 0, 0);

if (curItem != LB\_ERR)

{

TCHAR s[150];

MarkToDel(stream, curItem + 1);

ChangeMarkToDelete(stream, hDlg, hListBox, curItem + 1);

stream.seekg(0,stream.beg);

SendMessage(hListBox,LB\_RESETCONTENT,0,0);

for(AirBeg u; read(stream, u,0);)

{

swprintf(s, TEXT(" %-17d ║ %-17s ║ %-15s ║ %-15d ║ %-15d ║%d "), u.nom, u.marka, u.mesto, u.bilet, u.price,u.del);

SendMessage(hListBox,LB\_ADDSTRING,0,(LPARAM)s);

}

}

else

{

MessageBox(hDlg, L"сначала надо выделить элемент списка", L"ОШИБКА", MB\_OK);

break;

}

break;

}

case IDC\_DELETE:

DelMark(stream);

SendMessage(hListBox, LB\_RESETCONTENT, 0, 0);

ReadAll(stream, hListBox);

break;

case IDC\_REC\_ADD:

DialogBox(hInst, MAKEINTRESOURCE(IDD\_ADD\_REC), hDlg, GetDataDlgProc);

StructEnd itemU = obrabotkaInfo(airBeg);

SendMessage(hListBox, LB\_ADDSTRING, 0, (LPARAM)itemU.str);

add(stream, airBeg);

}

break;

}

return (INT\_PTR)FALSE;

}

INT\_PTR CALLBACK GetDataDlgProc(HWND hDlg, UINT message,WPARAM wParam, LPARAM lParam)

{

UNREFERENCED\_PARAMETER(lParam);

switch (message)

{

case WM\_INITDIALOG:

SetFocus(GetDlgItem(hDlg, IDC\_GET\_NOM));

return FALSE;

case WM\_COMMAND:

switch (LOWORD(wParam))

{

case IDOK:

airBeg = GetAir(hDlg);

EndDialog(hDlg, TRUE);

return TRUE;

case IDCANCEL:

airBeg.nom = 0;

\*airBeg.marka = 0;

\*airBeg.mesto = 0;

airBeg.bilet = 0;

airBeg.price = 0;

EndDialog(hDlg, FALSE);

return TRUE;

}

break;

}

return FALSE;

}

INT\_PTR CALLBACK PutDataDlgProc(HWND hDlg, UINT message, WPARAM wParam, LPARAM lParam)

{

UNREFERENCED\_PARAMETER(lParam);

static HWND hListBox;

static LOGFONT lf;

HFONT hFont1;

fstream stream(Filename, ios::in|ios::out);

if (!stream)

{

stream.close();

stream.open(Filename, ios::out);

stream.close();

stream.open(Filename, ios::in|ios::out);

}

stream.imbue(locale(".1251"));

switch (message)

{

case WM\_INITDIALOG:

hListBox = GetDlgItem(hDlg, IDC\_LIST1);

lf.lfHeight = 16;

lstrcpy((LPWSTR)&lf.lfFaceName, TEXT("Courier"));

hFont1 = CreateFontIndirect(&lf);

SendMessage(hListBox, WM\_SETFONT, (WPARAM)hFont1, TRUE);

SortedPrint(stream,hListBox);

return (INT\_PTR)TRUE;

case WM\_COMMAND:

if (LOWORD(wParam) == IDOK || LOWORD(wParam) == IDCANCEL)

{

EndDialog(hDlg, LOWORD(wParam));

return (INT\_PTR)TRUE;

}

break;

}

return (INT\_PTR)FALSE;

}

## 4.8 Файл “Files.cpp”

#include "stdafx.h"

#include "kursovayaSerkova.h"

extern AirBeg airBeg;

//+

void add(fstream& stream, AirBeg& airBeg, int n)

{

HWND hDlg;

stream.clear();

if (n)

{

stream.seekp((n - 1)\*LenRec, ios::beg);

}

else

{

stream.seekp(0, ios::end);

}

try

{

stream

<< setw(LenFam) << airBeg.nom

<< setw(LenSport) << airBeg.marka

<< setw(LenFam) << airBeg.mesto

<< setw(LenFam) << airBeg.bilet

<< setw(LenFam) << airBeg.price

<< setw(2) << airBeg.del

<< endl;

}

catch (...)

{

MessageBox(hDlg, TEXT("Файл не найден"), TEXT("Ошибка"), MB\_OK);

exit(0);

}

}

//+

fstream& read(fstream& stream, AirBeg &airBeg, int n)

{

stream.clear();

int k;

if (n)

{

k = stream.tellg();

stream.seekp((n - 1)\*LenRec, ios::beg);

k = stream.tellg();

}

stream >> setw(LenFam) >> airBeg.nom

>> setw(LenSport) >> airBeg.marka

>> setw(LenFam) >> airBeg.mesto

>> setw(LenFam) >> airBeg.bilet

>> setw(LenFam) >> airBeg.price

>> setw(2) >> airBeg.del;

if (!stream && !stream.eof())

{

exit(0);

}

return stream;

}

//+

void ReadAll(fstream& stream, HWND hListBox)

{

SendMessage(hListBox, LB\_RESETCONTENT, 0, 0);

stream.clear();

stream.seekg(0);

StructEnd startU;

while (read(stream, airBeg, 0))

{

startU = obrabotkaInfo(airBeg);

SendMessage(hListBox, LB\_ADDSTRING, 0, (LPARAM)startU.str);

}

}

//-

void MarkToDel(fstream& stream, int n)

{

stream.clear();

stream.seekg(0, ios::beg);

if (read(stream, airBeg, n))

{

airBeg.del = !airBeg.del;

}

add(stream, airBeg, n);

}

//+

void DelMark(fstream& stream)

{

fstream temp = fstream(Temp, ios::out);

temp.imbue(locale(".1251"));

stream.clear();

stream.seekg(0, ios::beg);

for (; read(stream, airBeg);)

{

if (airBeg.del == 0)

{

add(temp, airBeg);

}

}

temp.close();

stream.close();

remove(Filename);

rename(Temp, Filename);

stream.open(Filename, ios::in | ios::out);

}

//+

void ChangeMarkToDelete(fstream& stream, HWND hDlg, HWND hListBox, int curItem)

{

stream.clear();

stream.seekg(0);

read(stream, airBeg, curItem);

switch (airBeg.del)

{

case 0:

SetDlgItemText(hDlg, IDC\_POM, L"Нет");

break;

case 1:

SetDlgItemText(hDlg, IDC\_POM, L"Да");

break;

}

}

void SortedPrint(fstream& f,HWND hListBox)

{

AirBeg samolets[200]; // Буферны массив

int AmountOfsamolets = 0; // Кол-во самолетов

f.clear(); // Сброс потока

f.seekg(0, ios::beg);

for (int i = 1; read(f, samolets[i - 1], i); i++, AmountOfsamolets++); // Считываем все в массив самолёты

for (int i = 1; i < AmountOfsamolets; i++)

{

int j = i;

// Сравнение суммарной стоимости, для сортировки по убыванию

while ((samolets[j].bilet\*samolets[j].price)>(samolets[j-1].bilet\*samolets[j-1].price))

{

AirBeg buff = samolets[j - 1]; // Меняем местами

samolets[j - 1] = samolets[j];

samolets[j] = buff;

if (j - 1) j--; // Сдвигаемся с проверкой либо до первого элемента либо пока не будет отсортировано

else break;

}

}

TCHAR s[150];

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

{

swprintf(s, TEXT(" %-17s ║ %-15d"), samolets[i].marka, samolets[i].price);

SendMessage(hListBox,LB\_ADDSTRING,0,(LPARAM)s);

}

}

## 4.9 Файл “ObrabotkaInf.cpp”

#include "stdafx.h"

#include "kursovayaSerkova.h"

extern AirEnd studEnd;

AirBeg GetAir(HWND hDlg) {

AirBeg airBeg;

TCHAR buf[100] = TEXT("");

airBeg.del = 0;

BOOL lpNom = TRUE, lpMarka = TRUE, lpMesto = TRUE, lpBilet = TRUE, lpPrice = TRUE;

airBeg.nom = GetDlgItemInt(hDlg, IDC\_GET\_NOM, &lpNom, FALSE);

GetDlgItemText(hDlg, IDC\_GET\_MARKA, airBeg.marka, 12);

if (\*airBeg.marka == 0)

{

wcscpy(buf + wcslen(buf), TEXT(" ошибка ввода марки!"));

lpMarka = FALSE;

}

GetDlgItemText(hDlg, IDC\_GET\_MESTO, airBeg.mesto, 20);

if (\*airBeg.mesto == 0)

{

wcscpy(buf, TEXT(" ошибка ввода места"));

lpMesto = FALSE;

}

airBeg.bilet = GetDlgItemInt(hDlg, IDC\_GET\_BILET, &lpBilet, FALSE);

airBeg.price = GetDlgItemInt(hDlg, IDC\_GET\_PRICE, &lpPrice, FALSE);

if (lpNom == FALSE ||

lpMarka == FALSE ||

lpMesto == FALSE ||

lpBilet == FALSE ||

lpPrice == FALSE)

{

MessageBox(hDlg, buf, TEXT("Сообщение об ошибке ввода"), MB\_OK);

// установки признака пустой не введенной информации

airBeg.nom = 0;

\*airBeg.marka = 0;

\*airBeg.mesto = 0;

airBeg.bilet = 0;

airBeg.price = 0;

}

return airBeg;

}

void putStudEndToStroka(StructEnd& u)

{

swprintf(u.str, TEXT(" %-17d ║ %-17s ║ %-15s ║ %-15d ║ %-15d ║%d "), u.airEnd.nom, u.airEnd.marka, u.airEnd.mesto, u.airEnd.bilet, u.airEnd.price,u.airEnd.del);

//swprintf(u.str, TEXT(" %-17s ║ %-17u%% ║ %15u%% ║ %15u%% ║ %15u%% ║ "), u.airEnd.fam, u.airEnd.sport, u.airEnd.pob1, u.airEnd.pob2, u.airEnd.pob3 );

}

StructEnd obrabotkaInfo(AirBeg airBeg)

{

StructEnd u;

u.airEnd.nom = airBeg.nom;

wcscpy(u.airEnd.marka, airBeg.marka);

wcscpy(u.airEnd.mesto, airBeg.mesto);

u.airEnd.bilet = airBeg.bilet;

u.airEnd.price = airBeg.price;

u.airEnd.del = airBeg.del;

putStudEndToStroka(u);

return u;

}

## 4.10 Файл “stdafx.cpp”

// stdafx.cpp: исходный файл, содержащий только стандартные включаемые модули

// kursovayaSerkova.pch будет предкомпилированным заголовком

// stdafx.obj будет содержать предварительно откомпилированные сведения о типе

#include "stdafx.h"

// TODO: Установите ссылки на любые требующиеся дополнительные заголовки в файле STDAFX.H

// , а не в данном файле

# 5. Изображения работающей программы

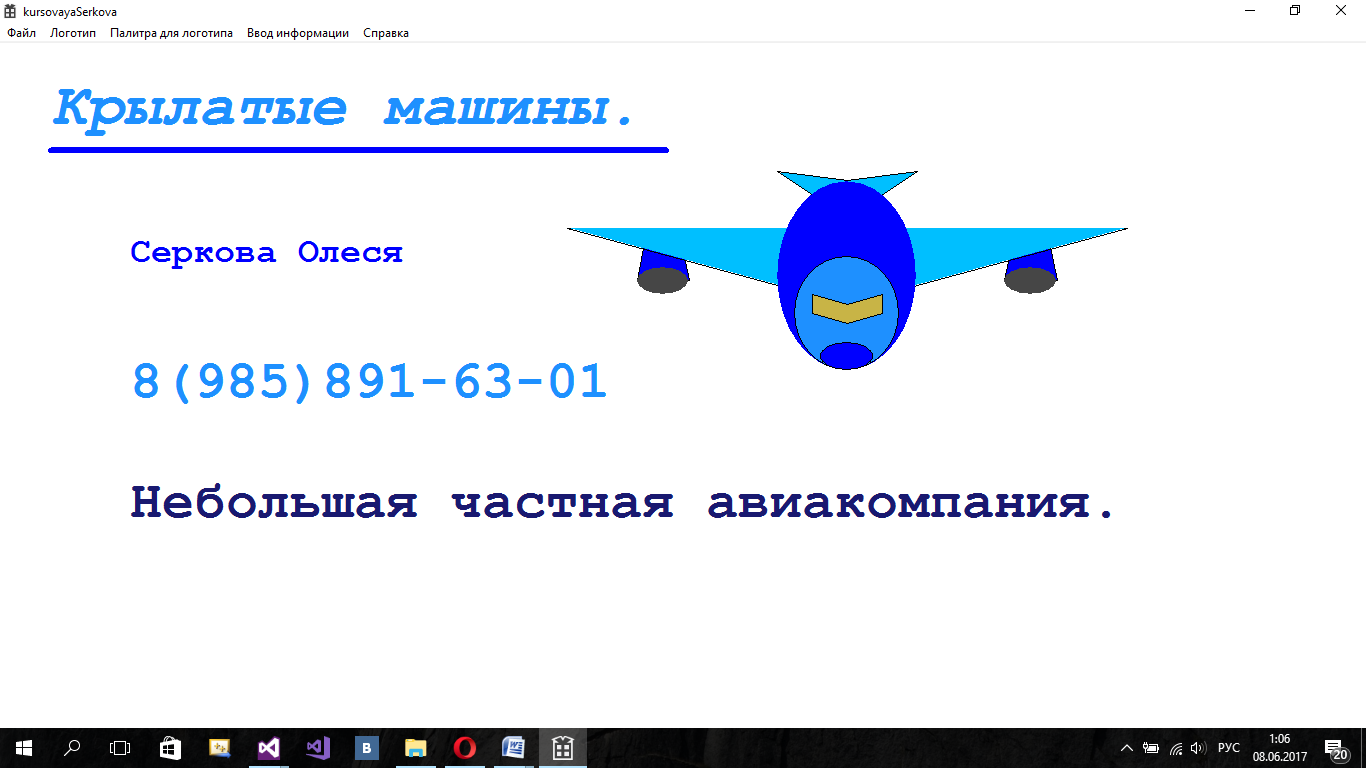


Рисунок 1 - Главное окно (вариант с текстом)

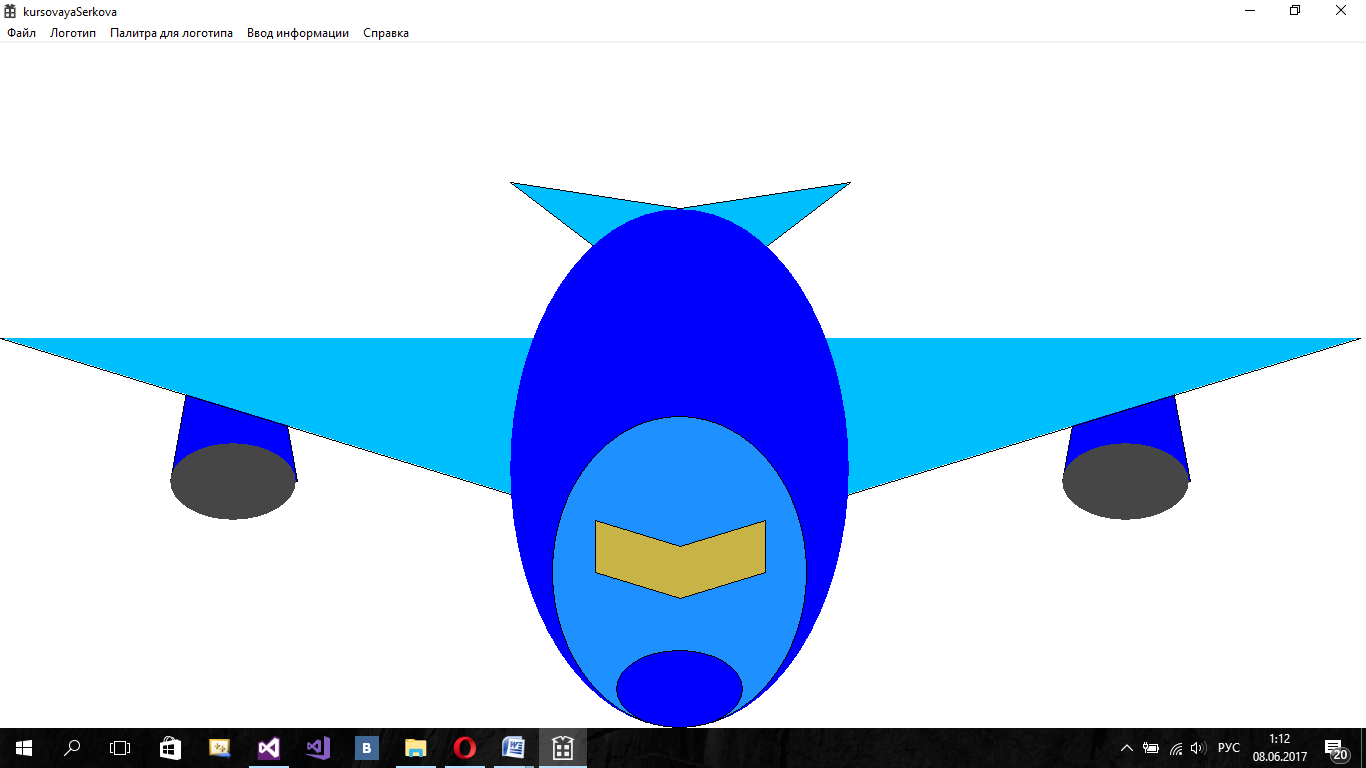


Рисунок 2 - Главное окно (вариант без текста)

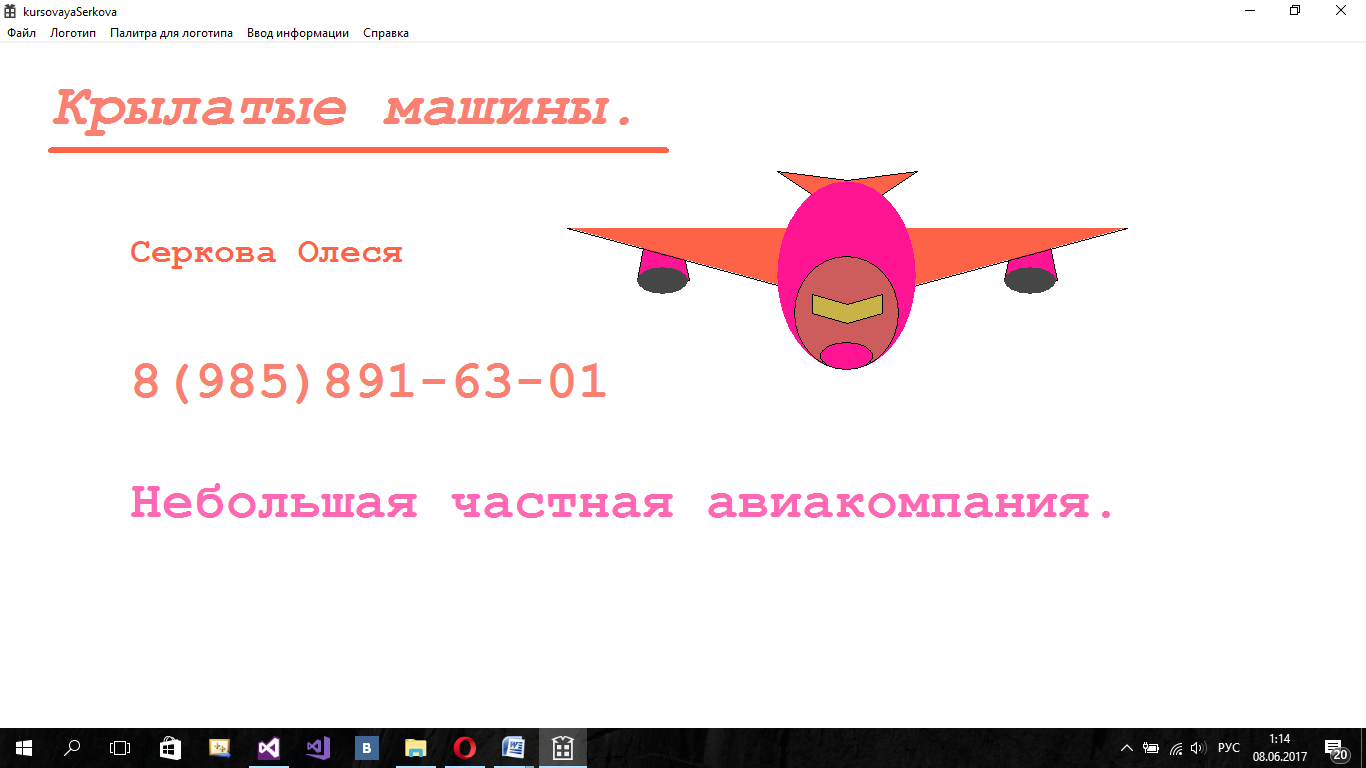


Рисунок 3 - Главное окно (Вторая цветовая гамма)

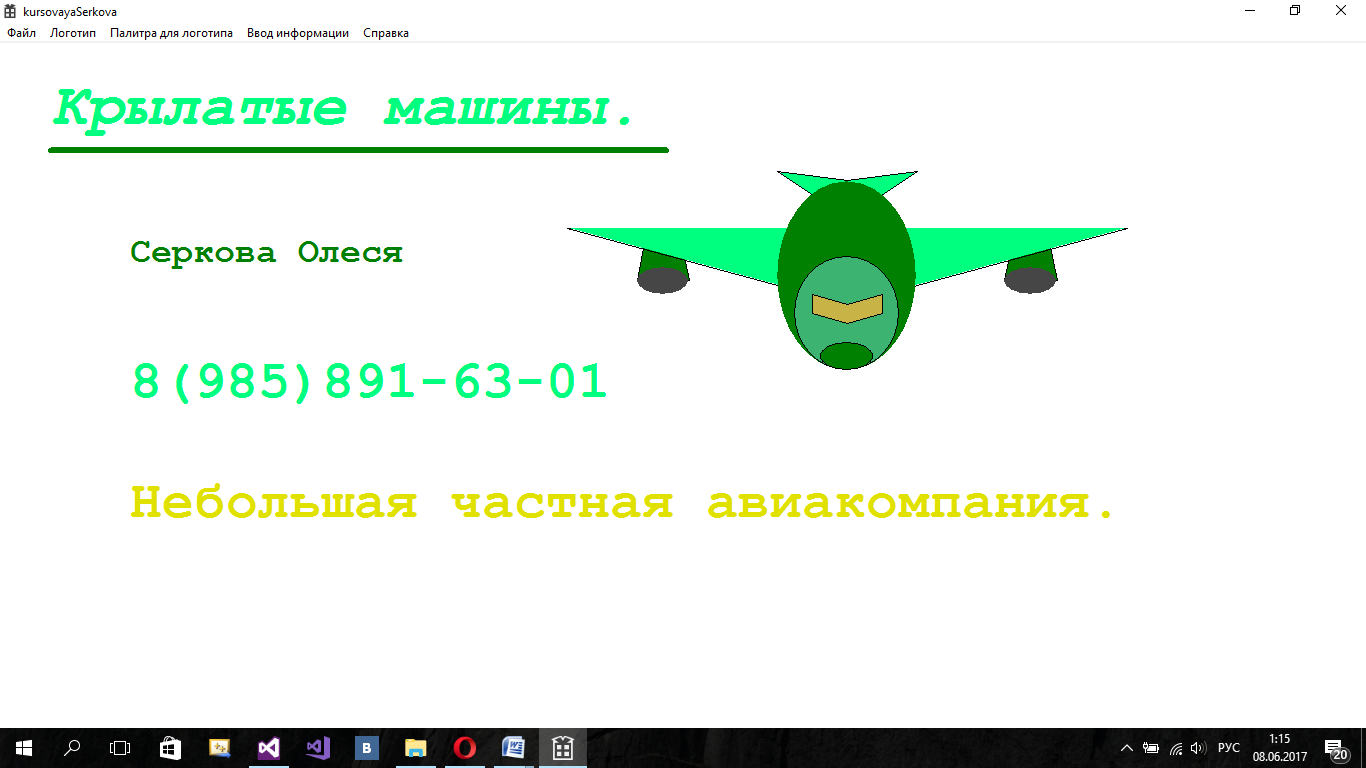


Рисунок 4 - Главное окно (Третья цветовая гамма)

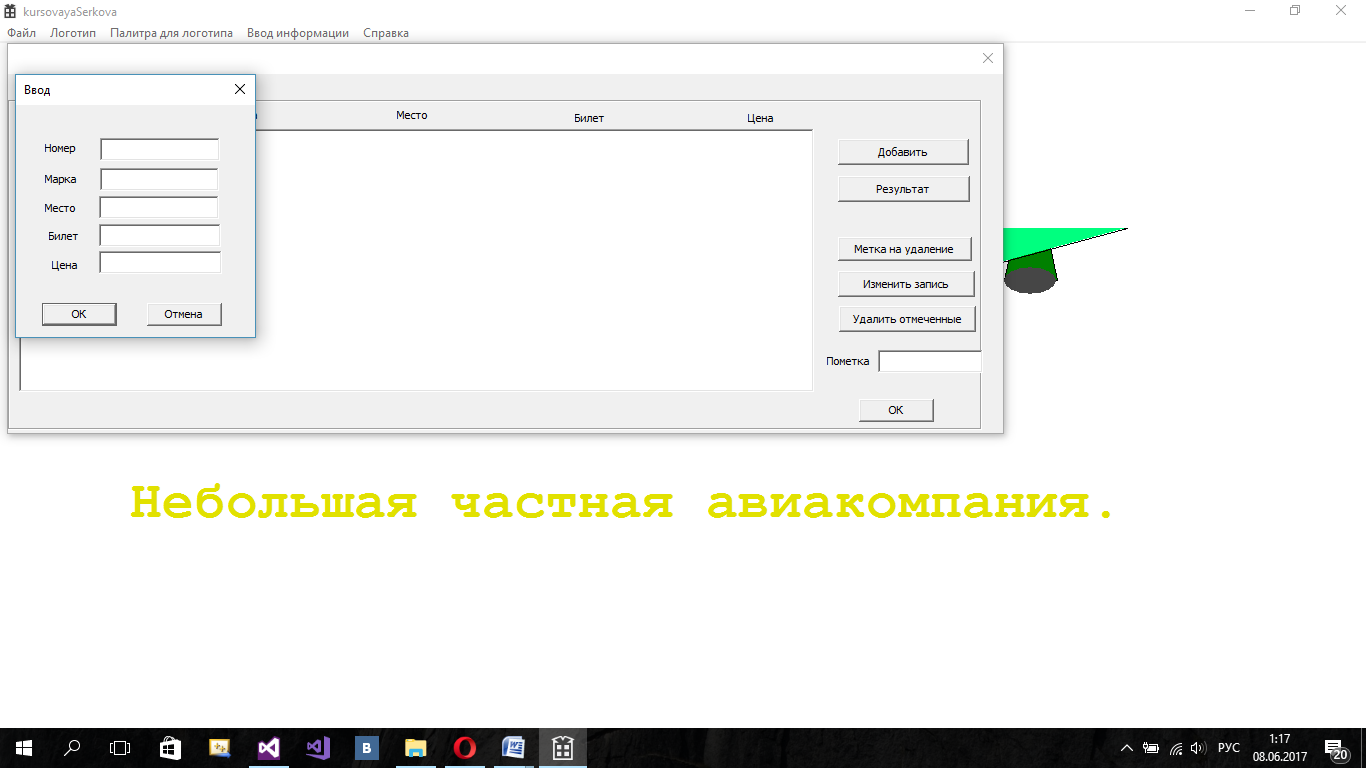


Рисунок 5 - Ввод информации

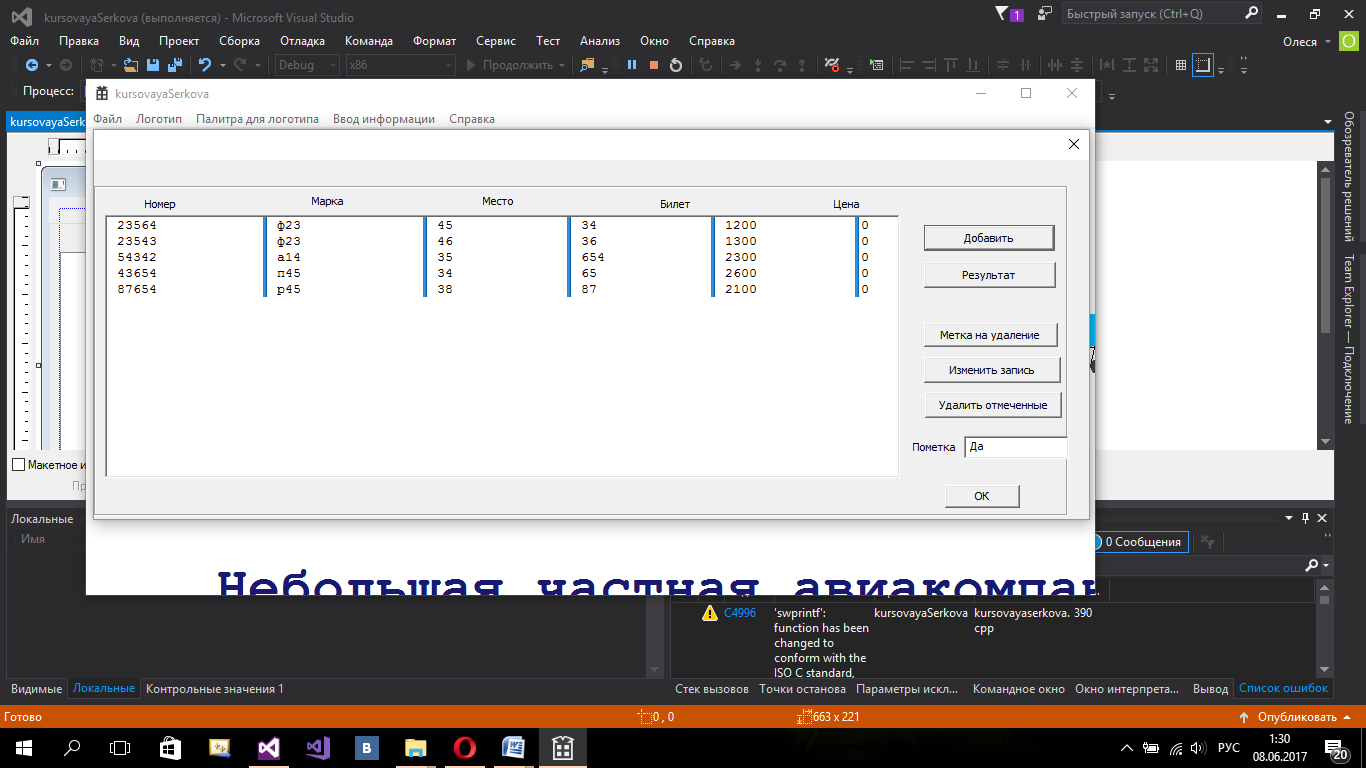


Рисунок 6 - Окно с информацией

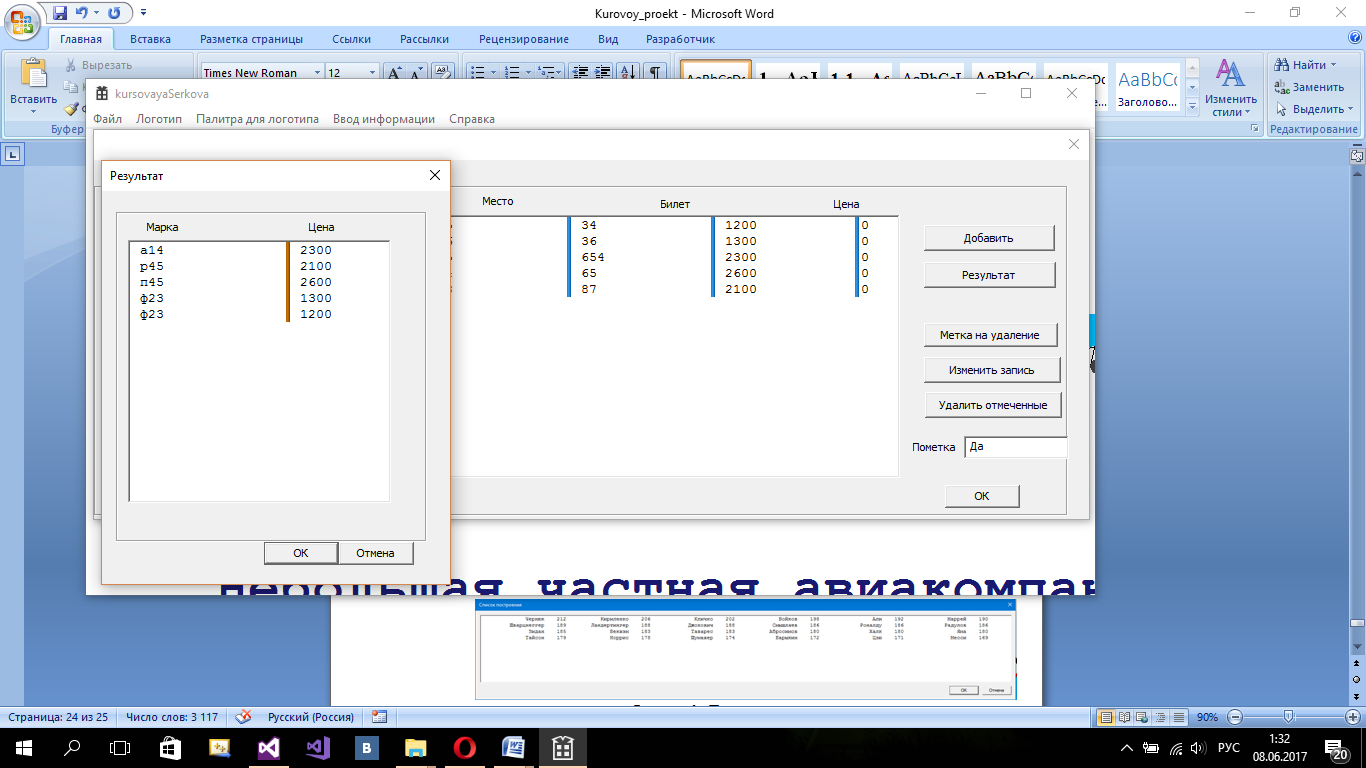


Рисунок 7 – Результат

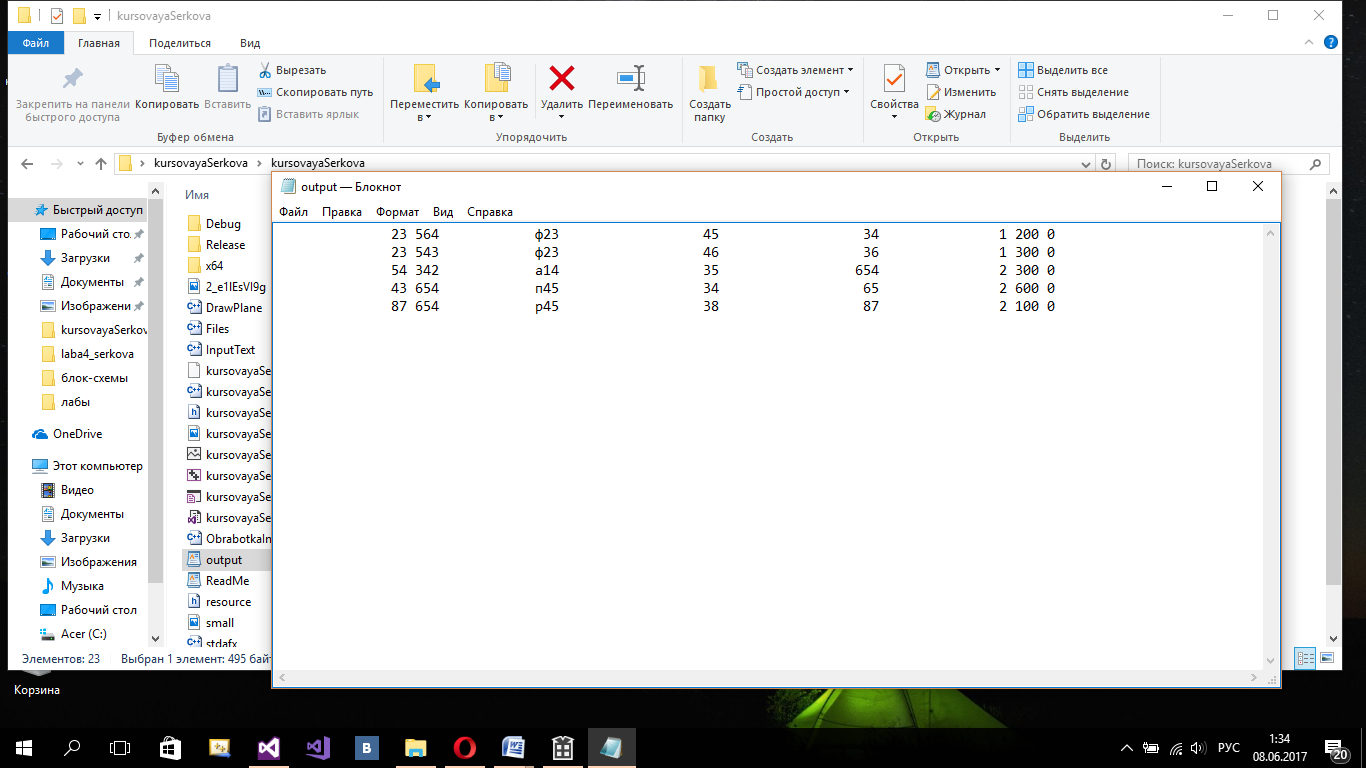


Рисунок 8 – Файл

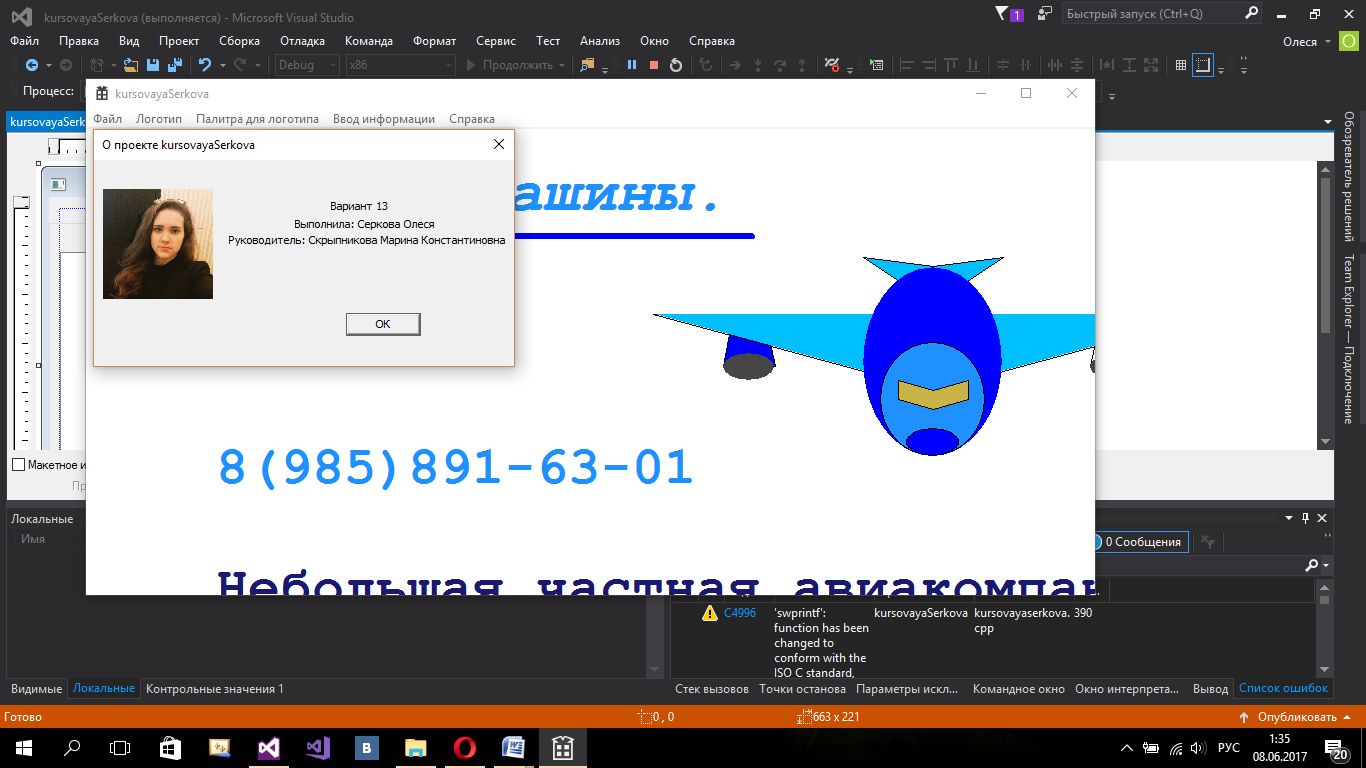


Рисунок 9 - Справка