МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ   
РОССИЙСКОЙ ФЕДЕРАЦИИ

**Федеральное государственное бюджетное образовательное учреждение**

**высшего образования**

**«Российский государственный университет им. А.Н. Косыгина   
(Технологии. Дизайн. Искусство)»**

Институт ИНФОРМАЦИОННЫЕ ТЕХНОЛОГИИ И ЦИФРОВЫЕ ТРАНСФОРМАЦИИ

**Отчет по лабораторной работе № 7**

**по дисциплине «Объектно-ориентированное программирование»**

**Тема: «Классы. Реализация класса авиарейсов»**

Выполнил: Сидоров Д. С., группа ИТС-123

Проверил: к.т.н., доц. Семёнов А. А.

Москва, 2025г

**Классы. Реализация класса авиарейсов**

Лабораторная работа посвящена изучению классов в ООП на C++. Задание:

Изучить и повторить проиллюстрированный ниже пример, демонстрирующий реализацию класса Reis, который положен в основу реализации списка записей авиарейсов. В примере разрабатывается визуальное приложение (Windows VCL Application), в котором необходимо создать класс с именем Reis, содержащий поля:

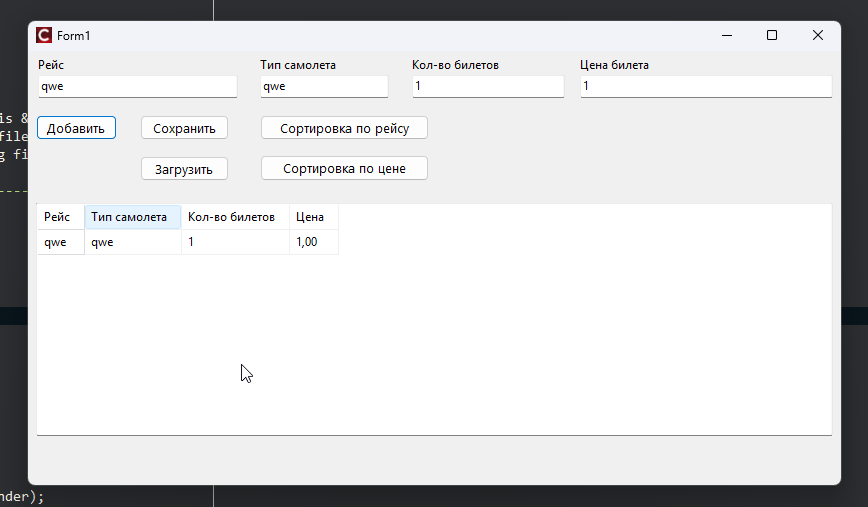
char name[N];

char type[N];

int kol;

float price;

В приложении реализовываются возможности добавления записей в таблицу StringGrid, записи их в текстовый файл, считывания оттуда и сортировки рейсов.

  
Рисунок 1. Демонстрация работы программы

**Листинг кода**

**Unit1.cpp**

////---------------------------------------------------------------------------

#include <vcl.h>

#include <vector>

#include <algorithm>

#pragma hdrstop

#include "Unit1.h"

//---------------------------------------------------------------------------

#pragma package(smart\_init)

#pragma resource "\*.dfm"

TForm1 \*Form1;

//---------------------------------------------------------------------------

\_\_fastcall TForm1::TForm1(TComponent\* Owner)

: TForm(Owner)

{

}

//---------------------------------------------------------------------------

void \_\_fastcall TForm1::ButtonAddClick(TObject \*Sender)

{

try {

Reis r(LE\_Reis->Text, LE\_Type->Text, StrToInt(LE\_Kol->Text), StrToFloat(LE\_Price->Text));

Reis::AddToGrid(StringGrid1, r);

} catch (...) {

ShowMessage(L"Ошибка ввода данных!");

}

}

//---------------------------------------------------------------------------

void Reis::AddToGrid(TStringGrid \*grid, const Reis &r)

{

int row = grid->RowCount;

grid->RowCount = row + 1;

grid->Cells[0][row] = r.name;

grid->Cells[1][row] = r.type;

grid->Cells[2][row] = IntToStr(r.kol);

grid->Cells[3][row] = FloatToStrF(r.price, ffFixed, 8, 2);

for (int col = 0; col < grid->ColCount; ++col) {

int textWidth = grid->Canvas->TextWidth(grid->Cells[col][row]) + 20;

if (textWidth > grid->ColWidths[col])

grid->ColWidths[col] = textWidth;

}

}

//---------------------------------------------------------------------------

void \_\_fastcall TForm1::ButtonSaveClick(TObject \*Sender)

{

if (SaveDialog1->Execute()) {

Reis::SaveToFile(StringGrid1, SaveDialog1->FileName);

}

}

//---------------------------------------------------------------------------

void Reis::SaveToFile(TStringGrid \*grid, String filename)

{

TStringList \*list = new TStringList();

for (int i = 1; i < grid->RowCount; i++) {

String line = grid->Cells[0][i] + "," +

grid->Cells[1][i] + "," +

grid->Cells[2][i] + "," +

grid->Cells[3][i];

list->Add(line);

}

list->SaveToFile(filename, TEncoding::UTF8);

delete list;

}

//---------------------------------------------------------------------------

void \_\_fastcall TForm1::ButtonLoadClick(TObject \*Sender)

{

if (OpenDialog1->Execute()) {

Reis::LoadFromFile(StringGrid1, OpenDialog1->FileName);

}

}

//---------------------------------------------------------------------------

void Reis::LoadFromFile(TStringGrid \*grid, String filename)

{

TStringList \*list = new TStringList();

list->LoadFromFile(filename, TEncoding::UTF8);

grid->RowCount = 1;

for (int i = 0; i < list->Count; ++i) {

TStringList \*cols = new TStringList();

cols->StrictDelimiter = true;

cols->Delimiter = ',';

cols->DelimitedText = list->Strings[i];

if (cols->Count == 4) {

try {

Reis r(cols->Strings[0], cols->Strings[1],

StrToInt(cols->Strings[2]), StrToFloat(cols->Strings[3]));

AddToGrid(grid, r);

} catch (...) {

// Пропустить строку с ошибкой

}

}

delete cols;

}

delete list;

}

//---------------------------------------------------------------------------

void \_\_fastcall TForm1::ButtonSortByNameClick(TObject \*Sender)

{

Reis::SortByName(StringGrid1);

}

//---------------------------------------------------------------------------

void \_\_fastcall TForm1::ButtonSortByPriceClick(TObject \*Sender)

{

Reis::SortByPrice(StringGrid1);

}

//---------------------------------------------------------------------------

void \_\_fastcall TForm1::FormCreate(TObject \*Sender)

{

StringGrid1->ColCount = 4;

StringGrid1->RowCount = 1;

StringGrid1->Cells[0][0] = L"Рейс";

StringGrid1->Cells[1][0] = L"Тип самолета";

StringGrid1->Cells[2][0] = L"Кол-во билетов";

StringGrid1->Cells[3][0] = L"Цена";

for (int col = 0; col < StringGrid1->ColCount; ++col) {

int maxWidth = StringGrid1->Canvas->TextWidth(StringGrid1->Cells[col][0]) + 20;

StringGrid1->ColWidths[col] = maxWidth;

}

SaveDialog1->Filter = L"Текстовые файлы (\*.txt)|\*.txt|Все файлы (\*.\*)|\*.\*";

OpenDialog1->Filter = L"Текстовые файлы (\*.txt)|\*.txt|Все файлы (\*.\*)|\*.\*";

}

//---------------------------------------------------------------------------

void Reis::SortByPrice(TStringGrid \*grid)

{

int rowCount = grid->RowCount;

std::vector<Reis> data;

for (int i = 1; i < rowCount; i++) {

data.emplace\_back(grid->Cells[0][i], grid->Cells[1][i],

StrToInt(grid->Cells[2][i]), StrToFloat(grid->Cells[3][i]));

}

std::sort(data.begin(), data.end(), [](const Reis &a, const Reis &b) {

return a.price < b.price;

});

grid->RowCount = 1;

for (const auto &r : data)

AddToGrid(grid, r);

}

//---------------------------------------------------------------------------

void Reis::SortByName(TStringGrid \*grid)

{

int rowCount = grid->RowCount;

std::vector<Reis> data;

for (int i = 1; i < rowCount; i++) {

data.emplace\_back(grid->Cells[0][i], grid->Cells[1][i],

StrToInt(grid->Cells[2][i]), StrToFloat(grid->Cells[3][i]));

}

std::sort(data.begin(), data.end(), [](const Reis &a, const Reis &b) {

return a.name < b.name;

});

grid->RowCount = 1;

for (const auto &r : data)

AddToGrid(grid, r);

}

**Unit1.h**

///---------------------------------------------------------------------------

#ifndef Unit1H

#define Unit1H

//---------------------------------------------------------------------------

#include <System.Classes.hpp>

#include <Vcl.Controls.hpp>

#include <Vcl.StdCtrls.hpp>

#include <Vcl.Forms.hpp>

#include <Vcl.Dialogs.hpp>

#include <Vcl.ExtCtrls.hpp>

#include <Vcl.Grids.hpp>

#include <Vcl.Mask.hpp>

#include <vector>

class Reis {

public:

String name;

String type;

int kol;

float price;

Reis(String n, String t, int k, float p)

: name(n), type(t), kol(k), price(p) {}

static void SortByName(TStringGrid \*grid);

static void SortByPrice(TStringGrid \*grid);

static void AddToGrid(TStringGrid \*grid, const Reis &r);

static void SaveToFile(TStringGrid \*grid, String filename);

static void LoadFromFile(TStringGrid \*grid, String filename);

};

//---------------------------------------------------------------------------

class TForm1 : public TForm

{

\_\_published: // IDE-managed Components

TStringGrid \*StringGrid1;

TButton \*ButtonAdd;

TButton \*ButtonSave;

TButton \*ButtonLoad;

TButton \*ButtonSortByName;

TButton \*ButtonSortByPrice;

TLabeledEdit \*LE\_Reis;

TLabeledEdit \*LE\_Type;

TLabeledEdit \*LE\_Kol;

TLabeledEdit \*LE\_Price;

TOpenDialog \*OpenDialog1;

TSaveDialog \*SaveDialog1;

void \_\_fastcall ButtonLoadClick(TObject \*Sender);

void \_\_fastcall ButtonSortByNameClick(TObject \*Sender);

void \_\_fastcall ButtonSortByPriceClick(TObject \*Sender);

void \_\_fastcall FormCreate(TObject \*Sender);

void \_\_fastcall ButtonAddClick(TObject \*Sender);

void \_\_fastcall ButtonSaveClick(TObject \*Sender);

private: // User declarations

public: // User declarations

\_\_fastcall TForm1(TComponent\* Owner);

};

//---------------------------------------------------------------------------

extern PACKAGE TForm1 \*Form1;

//---------------------------------------------------------------------------

#endif

**Вывод:** В ходе выполнения данной лабораторной работы был разработан класс Reis, реализующий объектную модель для представления и работы с информацией об авиарейсах. Созданный класс включает в себя необходимые поля для хранения названия рейса, типа самолета, количества мест и цены билета, что позволяет эффективно структурировать данные и работать с ними. В рамках разработки приложения были реализованы механизмы добавления записей в таблицу StringGrid, что обеспечивает наглядное представление информации о рейсах в удобном для пользователя формате. Особое внимание уделено функциональности сохранения данных в текстовый файл и последующего считывания из него, что демонстрирует работу с файловой системой и позволяет сохранять информацию между сеансами работы программы. Реализованный функционал сортировки рейсов по различным параметрам расширяет возможности работы с данными и улучшает пользовательский опыт. Разработка данного приложения позволила углубить понимание принципов объектно-ориентированного программирования в C++, в частности, механизмов инкапсуляции, работы с объектами класса и методами доступа к полям. Практическое применение полученных знаний в области ООП способствовало развитию навыков создания структурированных и расширяемых программных решений. Данная лабораторная работа является важным шагом в освоении объектно-ориентированного подхода к программированию и демонстрирует его преимущества при решении практических задач по организации и обработке структурированных данных.