**МИНОБРНАУКИ РОССИИ**

федеральное государственное автономное образовательное учреждение

высшего образования

«Санкт-Петербургский политехнический университет Петра Великого»

(ФГАОУ ВО «СПбПУ»)

Институт среднего профессионального образования

**ОТЧЕТ**

**по лабораторной работе №5**

по профессиональному модулю ПМ.01 «Разработка программных модулей программного обеспечения для компьютерных систем»

Специальность09.02.07 «Информационные системы и программирование»

Выполнили студенты 3 курса 32919/1 группы

Зориков Антон Алексеевич, Лейко Матвей Александрович

Преподаватель: Ильин Ю. П.

Санкт-Петербург

2024

**Cодержание**

[**Цель работы:** 3](#_Toc156485148)

[**Задание:** 3](#_Toc156485149)

[**Математическая формулировка методов:** 3](#_Toc156485150)

[**Текст программных модулей** 8](#_Toc156485151)

[**Протокол испытаний:** 19](#_Toc156485152)

[**Результаты выполнения работы программы** 21](#_Toc156485153)

**Тема: «Разработка класса по работе с бинарными файлами объектов данных»**

**Вариант №9**

# **Цель работы:**

Получить практические навыки создания программ обработки бинарных файлов, использующие методы социализации и десерилизации объектов с динамическими полями.

# **Задание:**

Создать структуру данных (класс) для отделения ОИТ, имеющую следующие поля: номер группы, фамилия имя отчество студента, год рождения, телефон.

Разработать следующие функции:

* добавление в файл объектов;
* вывод на экран содержимого всего файла
* корректировки отдельной записи, найденной по содержимому главного поля;
* отбор записей по содержимому одного из полей и вывод на экран найденных записей;

Интерфейс программы оформить в виде меню.

# **Математическая формулировка методов:**

1. Метод для получения добавления объектов.

public static void appendObject(string filename, Student student)

{

if (getPosition(filename, student.FIO) != -1)

{

throw new Exception("Запись с таким ключевым полем уже существует");

}

FileStream fa = new FileStream(filename, FileMode.Append);

BinaryFormatter bw = new BinaryFormatter();

bw.Serialize(fa, student);

fa.Close();

}

Данный метод принимает в качестве аргумента путь к файлу и объект студента.

1. Метод для нахождения позиции объекта в бинарном файле по его ключевому полю

public static int getPosition(string filename, string fio)

{

byte[] fioByte;

while (fio.Length < 60)

{

fio += " ";

}

if (fio.Length > 60) fio.Remove(60, fio.Length);

fioByte = Encoding.Unicode.GetBytes(fio);

int num = 0;

try

{

while (true)

{

using (FileStream fs = new FileStream(filename, FileMode.Open, FileAccess.ReadWrite))

{

BinaryFormatter bf = new BinaryFormatter();

fs.Seek(num \* 332, SeekOrigin.Begin);

Student student = (Student)bf.Deserialize(fs);

if (Enumerable.SequenceEqual(student.fioByte, fioByte))

{

return num;

}

}

num++;

}

}

catch (Exception e)

{

return -1;

}

}

Данный метод принимает на вход название файла и строку фио, возвращает номер позиции обьекта.

1. Метод для замены объектов в бинарном файле на другой.

public static void changeData(string filename, long num, Student st)

{

using (FileStream fs = new FileStream(filename, FileMode.Open)) { }

num = num < 0 ? 0 : num;

if (getPosition(filename, st.FIO) != -1)

{

throw new Exception("Запись с таким ключевым полем уже существует");

}

try

{

using (FileStream fs = new FileStream(filename, FileMode.Open, FileAccess.ReadWrite))

{

BinaryFormatter bf = new BinaryFormatter();

fs.Seek(num \* 332, SeekOrigin.Begin);

Student student = (Student)bf.Deserialize(fs);

student.Group = st.Group;

student.FIO = st.FIO;

student.BirthdayYear = st.BirthdayYear;

student.PhoneNumber = st.PhoneNumber;

student.fioByte = st.fioByte;

fs.Seek(num \* 332, SeekOrigin.Begin);

bf.Serialize(fs, student);

}

}

catch (Exception e)

{

throw new Exception(e.Message);

}

}

Данный метод принимает на вход название файла, позицию объекта в Файле и сам объект, метод ничего не возвращает. Производит замену объектов в файле.

1. Метод для создания файла

public static void createFile(string fileName)

{

FileStream stream = new FileStream(fileName, FileMode.OpenOrCreate);

stream.Close();

}

Данный метод принимает на вход строку с названием файла и создает или отрывает этот файл.

1. Метод для получения всех объектов в файле

public static List<Student> display(string fileName)

{

List<Student> students = new List<Student>();

try

{

BinaryFormatter formatter = new BinaryFormatter();

using (FileStream fs = new FileStream(fileName, FileMode.Open))

{

while (fs.Position < fs.Length)

{

Student student = (Student)formatter.Deserialize(fs);

students.Add(student);

}

}

}

catch (Exception e)

{

throw new Exception(e.Message);

}

return students;

}

Данный метод принимает на вход строку с именем файла и возвращает список всех студентов.

1. Метод для отбора студентов по группе

public static List<Student> getStudentsByGroup(string filename, ushort group)

{

List<Student> students = display(filename);

List<Student> result = new List<Student>();

foreach (Student student in students)

{

if (student.Group == group)

{

result.Add(student);

}

}

return result;

}

Данный метод принимает на вход строку с именем файла и номер группы. Возвращает список студентов соответствующей группы.

1. Метод для отбора студентов по номеру телефона

public static List<Student> getStudentsByPhoneNumber(string filename, uint phoneNumber)

{

List<Student> students = display(filename);

List<Student> result = new List<Student>();

foreach (Student student in students)

{

if (student.PhoneNumber == phoneNumber)

{

result.Add(student);

}

}

return result;

}Данный метод принимает на вход строку с именем файла и номер группы. Возвращает список студентов соответствующей группы.

1. Метод для отбора студентов по ФИО

public static List<Student> getStudentsByFIO(string filename, string fio)

{

List<Student> students = display(filename);

List<Student> result = new List<Student>();

byte[] fioByte;

while (fio.Length < 60)

{

fio += " ";

}

if (fio.Length > 60) fio.Remove(60, fio.Length);

fioByte = Encoding.Unicode.GetBytes(fio);

foreach (Student student in students)

{

if (Enumerable.SequenceEqual(student.fioByte, fioByte))

{

result.Add(student);

}

}

return result;

}

Данный метод принимает на вход строку с именем файла и номер группы. Возвращает список студентов соответствующей группы.

1. Метод для отбора студентов по году рождения

public static List<Student> getStudentsByBirthdayYear(string filename, ushort year)

{

List<Student> students = display(filename);

List<Student> result = new List<Student>();

foreach (Student student in students)

{

if (student.BirthdayYear == year)

{

result.Add(student);

}

}

return result;

}Данный метод принимает на вход строку с именем файла и номер группы. Возвращает список студентов соответствующей группы.

**Текст программных модулей:**

// библиотечный модуль \*.dll

using System;

using System.Collections.Generic;

using System.IO;

using System.Linq;

using System.Runtime.InteropServices;

using System.Runtime.Serialization;

using System.Runtime.Serialization.Formatters.Binary;

using System.Text;

using System.Text.RegularExpressions;

namespace OIT\_LIB

{

[Serializable]

public class Student

{

private ushort group;

public ushort Group

{

get { return group; }

set { group = value; }

}

[NonSerialized]

private string fio;

public string FIO

{

get { return fio; }

set

{

fio = value;

while (fio.Length < 60)

{

fio += " ";

}

if (fio.Length > 60) fio.Remove(60, fio.Length);

fioByte = Encoding.Unicode.GetBytes(fio);

}

}

private ushort birthdayYear { get; set; }

public ushort BirthdayYear

{

get { return birthdayYear; }

set

{

birthdayYear = value;

}

}

private ulong phoneNumber;

public ulong PhoneNumber

{

get { return phoneNumber; }

set { phoneNumber = value; }

}

public Student() { }

public Student(ushort group, string fio, ushort year, ulong phoneNum)

{

Group = group;

FIO = fio;

BirthdayYear = year;

PhoneNumber = phoneNum;

fioByte = Encoding.Unicode.GetBytes(FIO);

}

public byte[] fioByte;

}

public class OIT

{

public static void appendObject(string filename, Student student) //Метод для добавления обьекта в бинарный файл

{

if (getPosition(filename, student.FIO) != -1)

{

throw new Exception("Запись с таким ключевым полем уже существует");

}

FileStream fa = new FileStream(filename, FileMode.Append);

BinaryFormatter bw = new BinaryFormatter();

bw.Serialize(fa, student);

fa.Close();

}

public static int getPosition(string filename, string fio)//Метод для нахождения позиции объекта в бинарном файле по его ключевому полю

{

byte[] fioByte;

while (fio.Length < 60)

{

fio += " ";

}

if (fio.Length > 60) fio.Remove(60, fio.Length);

fioByte = Encoding.Unicode.GetBytes(fio);

int num = 0;

try

{

while (true)

{

using (FileStream fs = new FileStream(filename, FileMode.Open, FileAccess.ReadWrite))

{

BinaryFormatter bf = new BinaryFormatter();

fs.Seek(num \* 332, SeekOrigin.Begin);

Student student = (Student)bf.Deserialize(fs);

if (Enumerable.SequenceEqual(student.fioByte, fioByte))

{

return num;

}

}

num++;

}

}

catch (Exception e)

{

return -1;

}

}

public static void changeData(string filename, long num, Student st)//Метод для замены объектов в бинарном файле на другой.

{

using (FileStream fs = new FileStream(filename, FileMode.Open)) { }

num = num < 0 ? 0 : num;

if (getPosition(filename, st.FIO) != -1)

{

throw new Exception("Запись с таким ключевым полем уже существует");

}

try

{

using (FileStream fs = new FileStream(filename, FileMode.Open, FileAccess.ReadWrite))

{

BinaryFormatter bf = new BinaryFormatter();

fs.Seek(num \* 332, SeekOrigin.Begin);

Student student = (Student)bf.Deserialize(fs);

student.Group = st.Group;

student.FIO = st.FIO;

student.BirthdayYear = st.BirthdayYear;

student.PhoneNumber = st.PhoneNumber;

student.fioByte = st.fioByte;

fs.Seek(num \* 332, SeekOrigin.Begin);

bf.Serialize(fs, student);

}

}

catch (Exception e)

{

throw new Exception(e.Message);

}

}

public static void createFile(string fileName) //Метод для создания файла

{

FileStream stream = new FileStream(fileName, FileMode.OpenOrCreate);

stream.Close();

}

public static List<Student> display(string fileName)//Метод для получения всех объектов в файле

{

List<Student> students = new List<Student>();

try

{

BinaryFormatter formatter = new BinaryFormatter();

using (FileStream fs = new FileStream(fileName, FileMode.Open))

{

while (fs.Position < fs.Length)

{

Student student = (Student)formatter.Deserialize(fs);

students.Add(student);

}

}

}

catch (Exception e)

{

throw new Exception(e.Message);

}

return students;

}

public static List<Student> getStudentsByGroup(string filename, ushort group)// Метод для отбора студентов по группе

{

List<Student> students = display(filename);

List<Student> result = new List<Student>();

foreach (Student student in students)

{

if (student.Group == group)

{

result.Add(student);

}

}

return result;

}

public static List<Student> getStudentsByFIO(string filename, string fio)//Метод для отбора студентов по ФИО

{

List<Student> students = display(filename);

List<Student> result = new List<Student>();

byte[] fioByte;

while (fio.Length < 60)

{

fio += " ";

}

if (fio.Length > 60) fio.Remove(60, fio.Length);

fioByte = Encoding.Unicode.GetBytes(fio);

foreach (Student student in students)

{

if (Enumerable.SequenceEqual(student.fioByte, fioByte))

{

result.Add(student);

}

}

return result;

}

public static List<Student> getStudentsByBirthdayYear(string filename, ushort year)//Метод для отбора студентов по году рождения

{

List<Student> students = display(filename);

List<Student> result = new List<Student>();

foreach (Student student in students)

{

if (student.BirthdayYear == year)

{

result.Add(student);

}

}

return result;

}

public static List<Student> getStudentsByPhoneNumber(string filename, ulong phoneNumber)//Метод для отбора студентов по номеру телефона

{

List<Student> students = display(filename);

List<Student> result = new List<Student>();

foreach (Student student in students)

{

if (student.PhoneNumber == phoneNumber)

{

result.Add(student);

}

}

return result;

}

}

}

//код 1 формы

using System;

using System.Collections.Generic;

using System.Text;

using System.Windows.Forms;

using OIT\_LIB;

using static System.Windows.Forms.VisualStyles.VisualStyleElement;

namespace OITWindowsFormsApp

{

public partial class Form1 : Form

{

public Form1()

{

InitializeComponent();

}

private void открытьФормуПросмотраДанныхToolStripMenuItem\_Click(object sender, EventArgs e)

{

try

{

OIT.createFile(textBox4.Text);

ushort age;

ulong someValue;

if (ushort.TryParse(textBox1.Text, out age) && ulong.TryParse(textBox3.Text, out someValue))

{

Student student = new Student(age, textBox2.Text, (ushort)Convert.ToInt32(numericUpDown1.Value), someValue);

OIT.appendObject(textBox4.Text, student);

}

else

{

MessageBox.Show("Некорректный ввод данных", "Ошибка");

}

}

catch (Exception ex)

{

MessageBox.Show("Произошла ошибка: " + ex.Message, "Ошибка");

}

}

private void button1\_Click(object sender, EventArgs e)

{

try

{

OIT.createFile(textBox4.Text);

ushort age;

ulong someValue;

if (ushort.TryParse(textBox1.Text, out age) && ulong.TryParse(textBox3.Text, out someValue))

{

Student student = new Student(age, textBox2.Text, (ushort)Convert.ToInt32(numericUpDown1.Value), someValue);

OIT.appendObject(textBox4.Text, student);

}

else

{

MessageBox.Show("Некорректный ввод данных", "Ошибка");

}

}

catch (Exception ex)

{

MessageBox.Show("Произошла ошибка: " + ex.Message, "Ошибка");

}

}

private void button2\_Click(object sender, EventArgs e)

{

try

{

ushort age;

ulong someValue;

Student student;

if (ushort.TryParse(textBox1.Text, out age) && ulong.TryParse(textBox3.Text, out someValue))

{

student = new Student(age, textBox2.Text, (ushort)Convert.ToInt32(numericUpDown1.Value), someValue);

string filename = textBox4.Text;

OIT.changeData(filename, Convert.ToInt32(numericUpDown2.Value), student);

}

else

{

return;

}

}

catch (Exception ex)

{

MessageBox.Show($"Ошибка: {ex.Message}", "Ошибка");

}

}

private void button3\_Click(object sender, EventArgs e)

{

try

{

Form2 newForm = new Form2(textBox4.Text);

newForm.ShowDialog();

}

catch (Exception ex)

{

MessageBox.Show($"Ошибка: {ex.Message}", "Ошибка");

}

}

private void изменитьОбъектToolStripMenuItem\_Click(object sender, EventArgs e)

{

try

{

ushort age;

ulong someValue;

Student student;

if (ushort.TryParse(textBox1.Text, out age) && ulong.TryParse(textBox3.Text, out someValue))

{

student = new Student(age, textBox2.Text, (ushort)Convert.ToInt32(numericUpDown1.Value), someValue);

string filename = textBox4.Text;

OIT.changeData(filename, Convert.ToInt32(numericUpDown2.Value), student);

}

else

{

return;

}

}

catch (Exception ex)

{

MessageBox.Show($"Ошибка: {ex.Message}", "Ошибка");

}

}

private void открытьДанныеToolStripMenuItem\_Click(object sender, EventArgs e)

{

try {

Form2 newForm = new Form2(textBox4.Text);

newForm.ShowDialog();

}

catch (Exception ex)

{

MessageBox.Show($"Ошибка: {ex.Message}", "Ошибка");

}

}

private void textBox1\_TextChanged(object sender, EventArgs e)

{

}

private void textBox4\_TextChanged(object sender, EventArgs e)

{

}

private void textBox3\_TextChanged(object sender, EventArgs e)

{

}

}

}

// код 2 формы

using System;

using System.Collections.Generic;

using System.Data;

using System.IO;

using System.Windows.Forms;

using System.Collections.Generic;

using OIT\_LIB;

using System.Text;

namespace OITWindowsFormsApp

{

public partial class Form2 : Form

{

private string data;

public Form2(string data)

{

InitializeComponent();

this.data = data;

}

private void Form2\_Load(object sender, EventArgs e)

{

try

{

List<Student> students = OIT.display(data);

if (students.Count == 0)

{

MessageBox.Show("Записи не найдены", "Ошибка");

return;

}

dataGridView1.RowCount = students.Count;

dataGridView1.ColumnCount = 5;

for (int i = 0; i < students.Count; i++)

{

Student student = students[i];

dataGridView1.Rows[i].Cells[0].Value = i;

dataGridView1.Rows[i].Cells[1].Value = student.Group;

string str = "";

char[] bytes = Encoding.UTF8.GetString(student.fioByte).ToCharArray();

foreach (char c in bytes) { str += c; }

dataGridView1.Rows[i].Cells[2].Value = str;

dataGridView1.Rows[i].Cells[3].Value = student.BirthdayYear;

dataGridView1.Rows[i].Cells[4].Value = student.PhoneNumber;

}

}

catch (Exception ex)

{

MessageBox.Show($"Ошибка: {ex.Message}", "Ошибка");

}

}

private void UpdateDataGridView()

{

dataGridView1.DataSource = null;

}

private void FilterData()

{

int selectedIndex = comboBox1.SelectedIndex;

List<Student> students = new List<Student>();

bool exit = false;

switch (selectedIndex)

{

case 0:

students = OIT.getStudentsByGroup(data, Convert.ToUInt16(textBox4.Text));

break;

case 1:

students = OIT.getStudentsByFIO(data, textBox4.Text);

break;

case 3:

students = OIT.getStudentsByPhoneNumber(data, Convert.ToUInt64(textBox4.Text));

break;

case 2:

students = OIT.getStudentsByBirthdayYear(data, Convert.ToUInt16(textBox4.Text));

// MessageBox.Show("Значение не может быть преобразовано в число", "Ошибка");

break;

}

if (exit) { return; }

if (students.Count == 0)

{

MessageBox.Show("Записи не найдены", "Ошибка");

return;

}

dataGridView1.RowCount = students.Count;

dataGridView1.ColumnCount = 5;

for (int i = 0; i < students.Count; i++)

{

Student student = students[i];

dataGridView1.Rows[i].Cells[0].Value = i;

dataGridView1.Rows[i].Cells[1].Value = student.Group;

string str = "";

char[] bytes = Encoding.UTF8.GetString(student.fioByte).ToCharArray();

foreach (char c in bytes) { str += c; }

dataGridView1.Rows[i].Cells[2].Value = str;

dataGridView1.Rows[i].Cells[3].Value = student.BirthdayYear;

dataGridView1.Rows[i].Cells[4].Value = student.PhoneNumber;

}

}

private void button1\_Click(object sender, EventArgs e)

{

if (File.Exists(data))

{

FilterData();

}

else

{

MessageBox.Show("Файл не найден.");

}

}

private void button2\_Click(object sender, EventArgs e)

{

try

{

List<Student> students = OIT.display(data);

if (students.Count == 0)

{

MessageBox.Show("Записи не найдены", "Ошибка");

return;

}

dataGridView1.RowCount = students.Count;

dataGridView1.ColumnCount = 5;

for (int i = 0; i < students.Count; i++)

{

Student student = students[i];

dataGridView1.Rows[i].Cells[0].Value = i;

dataGridView1.Rows[i].Cells[1].Value = student.Group;

string str = "";

char[] bytes = Encoding.UTF8.GetString(student.fioByte).ToCharArray();

foreach (char c in bytes) { str += c; }

dataGridView1.Rows[i].Cells[2].Value = str;

dataGridView1.Rows[i].Cells[3].Value = student.BirthdayYear;

dataGridView1.Rows[i].Cells[4].Value = student.PhoneNumber;

}

}

catch (Exception ex)

{

MessageBox.Show($"Ошибка: {ex.Message}", "Ошибка");

}

}

}

}

# **Протокол испытаний:**

*Таблица 1.* Протокол испытаний

|  |  |  |  |
| --- | --- | --- | --- |
| **Проверяемые требования** | **Сообщения программы и вводимые значения** | **Ожидаемые результаты** | **Фактические результаты** |
| Добавление объекта с корректными данными | Номер группы «319»  ФИО «Zorikov A. A.»  Год рождения «2005»  Телефон «9118902367»  Имя файла «1.dat» | Успешное добавление объекта в таблицу | Успешное добавление объекта в таблицу |
| Проверка неверного ввода номера группы. | Номер группы «319i»  ФИО «Zorikov A. A.»  Год рождения «2005»  Телефон «9118902367»  Имя файла «1.dat» | Сообщение об ошибке “Некорректный ввод данных” | Сообщение об ошибке “Некорректный ввод данных” |
| Проверка неверного ввода телефона. | Номер группы «319»  ФИО «Zorikov A. A.»  Год рождения «2005»  Телефон «911890236743567894567894567856756767678»  Имя файла «1.dat» | Сообщение об ошибке “Некорректный ввод данных” | Сообщение об ошибке “Некорректный ввод данных” |
| Проверка добавления данных с одинаковым ключевым полем. | Номер группы «319»  ФИО «Zorikov A. A.»  Год рождения «2005»  Телефон «9118902367»  Имя файла «1.dat» | Сообщение об ошибке “Запись с таким ключевым полем уже существует” | Сообщение об ошибке “Запись с таким ключевым полем уже существует” |
| Проверка замена данных. | Номер группы «319»  ФИО «Zorikov A. A. \_1»  Год рождения «2005»  Телефон «9118902367»  Имя файла «1.dat» | Успешное добавление объекта в таблицу | Успешное добавление объекта в таблицу |
| Проверка замена данных на данные с совпадающим ключевым полем. | Номер группы «319»  ФИО «Zorikov A. A.»  Год рождения «2005»  Телефон «9118902367»  Имя файла «1.dat» | Сообщение об ошибке “Запись с таким ключевым полем уже существует” | Сообщение об ошибке “Запись с таким ключевым полем уже существует” |
| Проверка открытия данных без ввод названия файла | Номер группы «319»  ФИО «Zorikov A. A.»  Год рождения «2005»  Телефон «9118902367»  Имя файла «» | Сообщение об ошибке “Пустое имя пути не допускается” | Сообщение об ошибке “Пустое имя пути не допускается” |
| Отбор по номеру группы | Выбор отбора «Отбор по группе»  Введите данные «319» | Успешный вывод отобранных по группе данных в виде таблицы. | Успешный вывод отобранных по группе данных в виде таблицы. |
| Отбор по ФИО | Выбор отбора «Отбор по ФИО»  Введите данные «319» | Успешный вывод отобранных по ФИО данных в виде таблицы. | Успешный вывод отобранных по ФИО данных в виде таблицы. |
| Отбор по году рождения | Выбор отбора «Отбор по году рождения»  Введите данные «319» | Успешный вывод отобранных по году рождения данных в виде таблицы. | Успешный вывод отобранных по году рождения данных в виде таблицы. |
| Отбор по номеру телефона | Выбор отбора «Отбор по номеру телефона»  Введите данные «319» | Успешный вывод отобранных по номеру телефона данных в виде таблицы. | Успешный вывод отобранных по номеру телефона данных в виде таблицы. |
| Вывод всех данных. | Нажать на кнопку вывести все | Успешный вывод всех данных из файла. | Успешный вывод всех данных из файла. |

**Результаты выполнения работы программы:**

1. Добавление объекта с корректными данными.

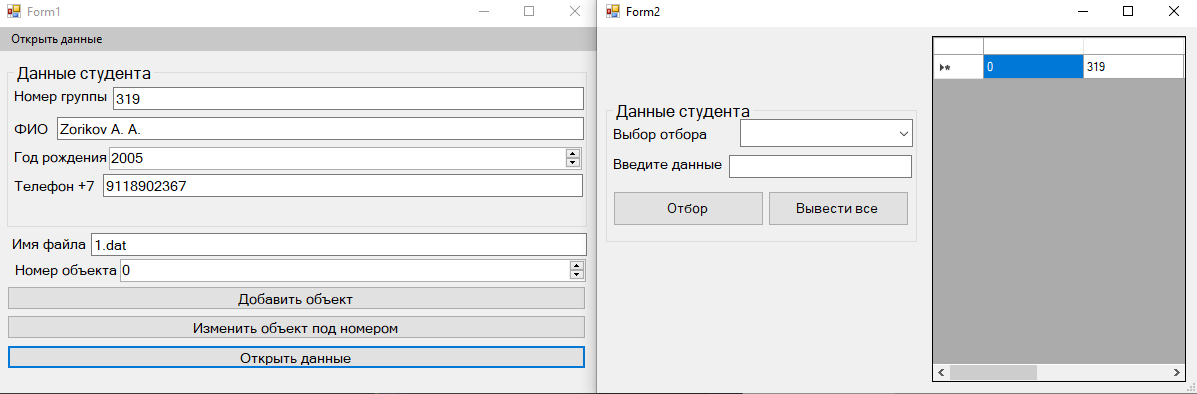


рис.1 Успешное добавление объекта в таблицу

1. Проверка неверного ввода номера группы.

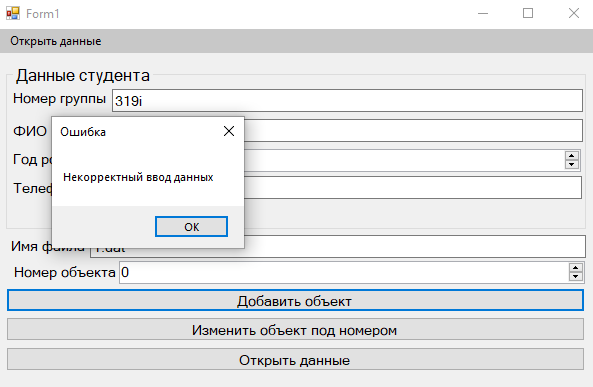


рис.2 Сообщение об ошибке “Сообщение об ошибке “Некорректный ввод данных””

1. Проверка неверного ввода телефона.

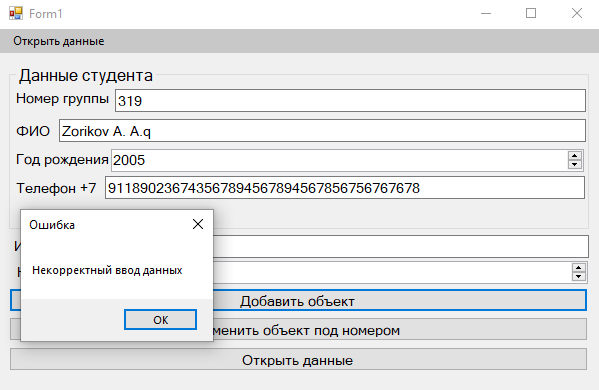


рис.3 Сообщение об ошибке “Сообщение об ошибке “Некорректный ввод данных””

1. Проверка добавления данных с одинаковым ключевым полем.

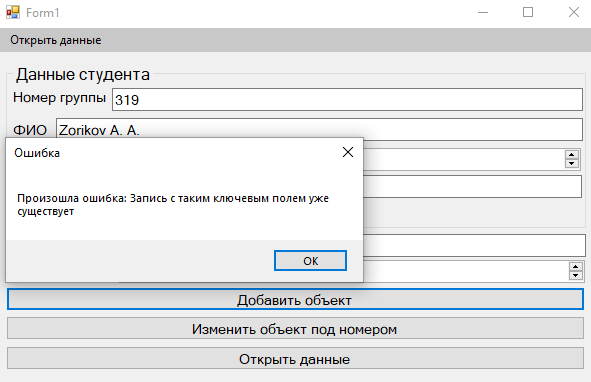


рис.4 Сообщение об ошибке “Запись с таким ключевым полем уже существует”

1. Проверка замена данных.

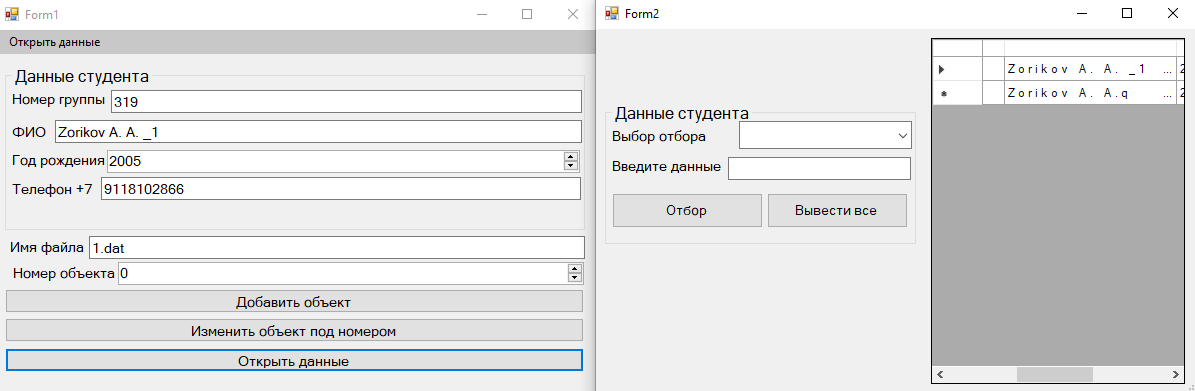


рис.5 Успешное добавление объекта в таблиц

1. Проверка замена данных на данные с совпадающим ключевым полем.

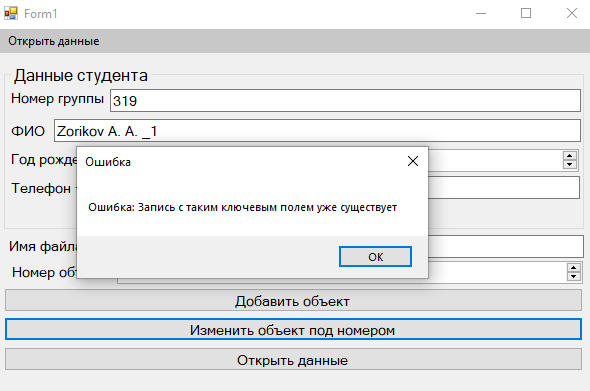


рис.6 Сообщение об ошибке “Запись с таким ключевым полем уже существует”

1. Проверка открытия данных без ввод названия файла

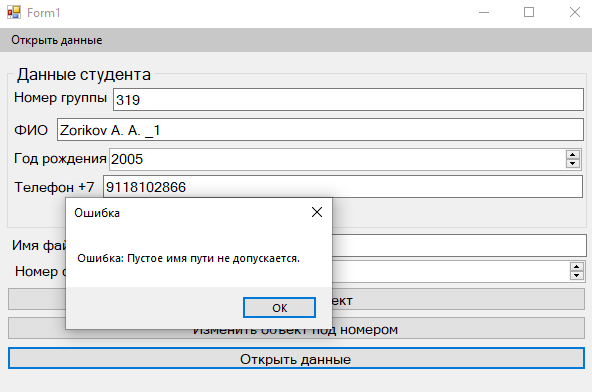


рис.7 Сообщение об ошибке “Пустое имя пути не допускается”

1. Отбор по номеру группы

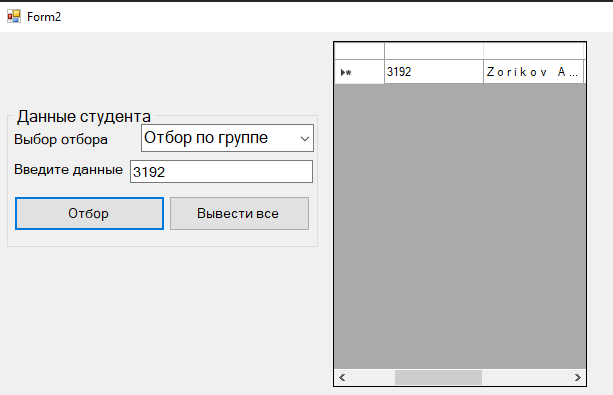


рис.8 Успешный вывод отобранных по группе данных в виде таблицы.

1. Отбор по ФИО

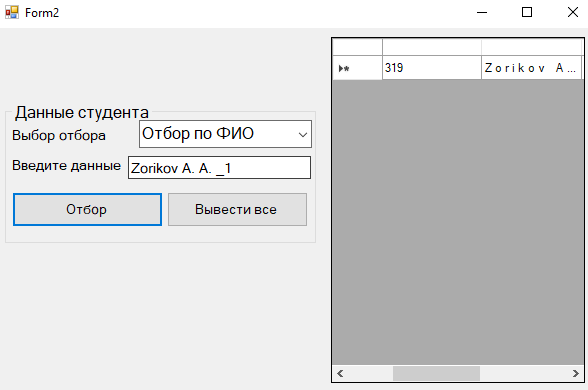


рис.9 Успешный вывод отобранных по группе данных в виде таблицы.

1. Отбор по году рождения

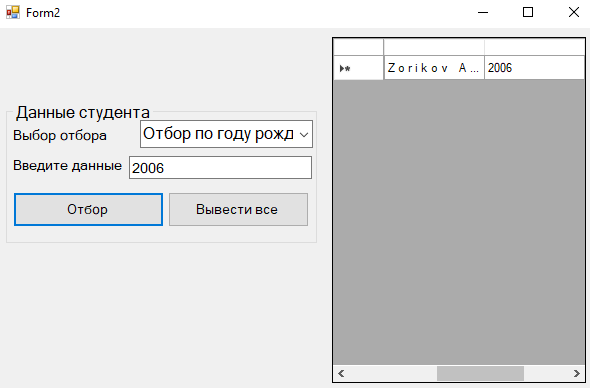


рис.10 Успешный вывод отобранных по году рождения данных в виде таблицы.

1. Отбор по номеру телефона

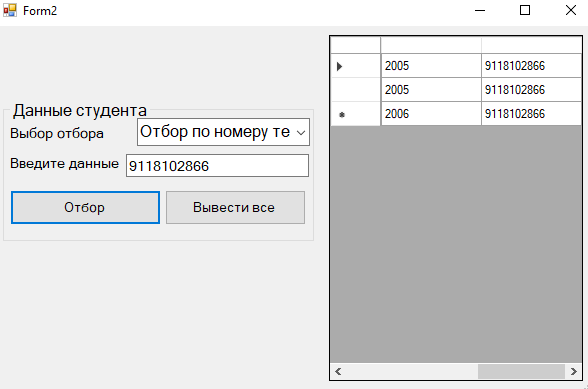


рис.11 Успешный вывод отобранных по группе данных в виде таблицы.

1. Вывод всех данных.

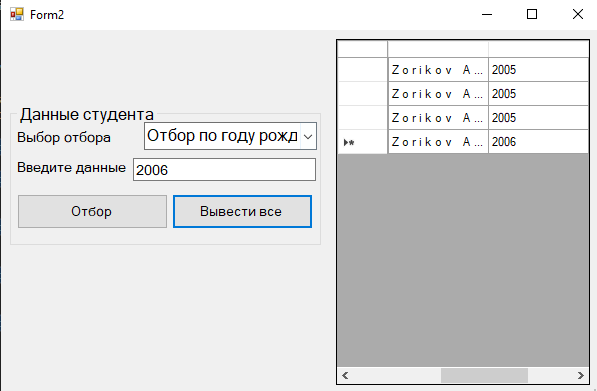


рис.12 Успешный вывод всех данных из файла.