Додаток А – код програми

```
public partial class WorkersForm : Form
    {
        //таблица данных
        public DataTable table = new DataTable();
        public WorkerCollection toSave;
        //конструктор формы с аргументом - отображаемой коллекции
        public WorkersForm(WorkerCollection col)
        {
            InitializeComponent();
            toSave = new WorkerCollection();
            table.Columns.Add("№");
            table.Columns.Add("�мя");
            table.Columns.Add("Фамилия");
            table.Columns.Add("Дата наема");
            table.Columns.Add("Должность");
            toSave = col;
            DisplayList();

        }

        private void WorkersForm_Load(object sender, EventArgs e)
        {
            //WorkerView.DataMember = new Worker();
        }

        //метод отображения списка
        private void DisplayList()
        {

            table.Rows.Clear();
            int index = 0;

            foreach (Worker worker in toSave)
            {
                DisplayWorker(worker, index);
                index++;
            }
            WorkerView.DataSource = table;
            WorkerView.Columns[0].Visible = false;
        }

        //метод отображения строки
        private void DisplayWorker(Worker worker, int index)
        {
            DataRow row = table.NewRow();
            row["№"] = index;
            row["�мя"] = worker.FirstName;
            row["Фамилия"] = worker.SecondName;
            row["Дата наема"] = worker.DateOfStart.Day + "/" +
worker.DateOfStart.Month + "/" + worker.DateOfStart.Year;
            row["Должность"] = worker.Position;
            table.Rows.Add(row);
        }

        private void WorkerView_CellContentClick(object sender,
DataGridViewCellEventArgs e)
        {

        }

        //метод открытия формы добавления
        private void AddButton_Click(object sender, EventArgs e)
        {
            WorkerAdd form = new WorkerAdd(toSave);
            form.ShowDialog();
            DisplayList();
        }

        //метод открытия формы редактирования
        private void EditButton_Click(object sender, EventArgs e)
        {
            int index =
Convert.ToInt32(WorkerView.SelectedCells[0].OwningRow.Cells[0].Value);
            WorkerAdd form = new WorkerAdd(toSave,index);
            form.ShowDialog();
            DisplayList();
        }

        //метод удаления выбраного элемента
        private void DeleteButton_Click(object sender, EventArgs e)
        {
            int index =
Convert.ToInt32(WorkerView.SelectedCells[0].OwningRow.Cells[0].Value);
            toSave.RemoveAt(index);
            DisplayList();
        }

        //метод поиска в списке
        private void SearchTextBox_TextChanged(object sender, EventArgs e)
        {
            string query = SearchTextBox.Text;
            if (query == "")
            {
                DisplayList();
            }
            else
            {
                table.Rows.Clear();
                int index = 0;
                foreach (Worker wrkr in toSave)
                {
                    if (wrkr.FirstName.ToLower().IndexOf(query.ToLower()) !=
-1 || wrkr.SecondName.ToLower().IndexOf(query.ToLower()) != -1 ||
wrkr.Position.ToLower().IndexOf(query.ToLower()) != -1)
                    {
                        DisplayWorker(wrkr, index);
                    }
                    index++;
                }

            }
        }

        private void CloseButton_Click(object sender, EventArgs e)
        {
            Close();
        }
    }


    public partial class WorkerAdd : Form
    {
        //�ндекс, если передается в форму
        private int Index;
        //Внешний список
        private WorkerCollection List;
        //флаг редактирования
        private bool edit;

        //конструктор добавления
        public WorkerAdd(WorkerCollection list)
        {
            InitializeComponent();
            edit = false;
            List = list;

        }

        //конструктор редактирования
        public WorkerAdd(WorkerCollection list, int index)
        {
            InitializeComponent();
            Index = index;
            edit = true;
            List = list;
            FirstName.Text = list[index].FirstName;
            SecondName.Text = list[index].SecondName;
            Position.Text = list[index].Position;
            Day.Text = list[index].DateOfStart.Day.ToString();
            Month.Text = list[index].DateOfStart.Month.ToString();
            Year.Text = list[index].DateOfStart.Year.ToString();
        }

        //отработка кнопки добавления
        private void Add_button_Click(object sender, EventArgs e)
        {
            int day = Convert.ToInt32(Day.Text);
            int month = Convert.ToInt32(Month.Text);
            int year = Convert.ToInt32(Year.Text);
            if (day > 31 || day < 1 || month > 12 || month < 1 ||
FirstName.Text == "" || SecondName.Text == "" || Position.Text == "")
            {
                MessageBox.Show("Проверьте правильность ввода");
                return;
            }
            if (edit)
            {
                List[Index] = new Worker(FirstName.Text, SecondName.Text,
Position.Text,
                    new Date(day, month, year));
            }
            else
            {
                List.Add(new Worker(FirstName.Text, SecondName.Text,
Position.Text,
                    new Date(day, month, year)));
            }
            Close();
        }

        private void cancelButton_Click(object sender, EventArgs e)
        {
            Close();
        }
    }


    public partial class searchForm : Form
    {
        public searchForm()
        {
            InitializeComponent();
            table.Columns.Add("№");
            table.Columns.Add("�мя");
            table.Columns.Add("Фамилия");
            table.Columns.Add("Камера");
            table.Columns.Add("Статья");
            table.Columns.Add("Дата заключения");
            table.Columns.Add("Место в тюремной иерархии");
            table.Columns.Add("Родственики");
            table.Columns.Add("Характер");
            DisplayList();
        }
        DataTable table = new DataTable();
        Prison jailed = new Prison();

        private void Search_Load(object sender, EventArgs e) { }

        private void DisplayList()
        {

            table.Rows.Clear();
            int index = 0;


            foreach (Prisoner prisoner in jailed)
            {
                DisplayPrisoner(prisoner, index);
                index++;
            }
            tablePrison.DataSource = table;
            tablePrison.Columns[0].Visible = false;
        }

        private void DisplayPrisoner(Prisoner prisoner, int index)
        {
            DataRow row = table.NewRow();
            row["№"] = index;
            row["�мя"] = prisoner.First_Name;
            row["Фамилия"] = prisoner.Second_Name;
            row["Камера"] = prisoner.Room;
            row["Статья"] = prisoner.Criminality;
            row["Дата заключения"] = prisoner.DateOfJailed.Day + "/" +
prisoner.DateOfJailed.Month + "/" + prisoner.DateOfJailed.Year;
            row["Место в тюремной иерархии"] = prisoner.PITPH;
            if (prisoner.Relatives == "")
            {
                row["Родственики"] = "-";
            }
            else
            {
                row["Родственики"] = prisoner.Relatives;
            }
            row["Характер"] = prisoner.Temper;
            table.Rows.Add(row);

        }
        private void Search()
        {
            string query = searchTextBox.Text;
            if (query == "")
            {
                DisplayList();
            }
            else
            {
                table.Rows.Clear();
                int index = 0;
                foreach (Prisoner prisoner in jailed)
                {
                    if
(prisoner.First_Name.ToUpper().IndexOf(query.ToUpper()) != -1 ||
prisoner.Second_Name.ToUpper().IndexOf(query.ToUpper()) != -1 ||
Convert.ToString(prisoner.Room).IndexOf(query) == 0 ||
prisoner.Criminality.ToUpper().IndexOf(query.ToUpper()) == 0 ||/*
prisoner.DateOfJailed.Day == 0 || prisoner.DateOfJailed.Month == 0 ||
prisoner.DateOfJailed.Month == 0 */
prisoner.PITPH.ToUpper().IndexOf(query.ToUpper()) != -1 ||
prisoner.Relatives.ToUpper().IndexOf(query.ToUpper()) != -1 ||
prisoner.Temper.ToUpper().IndexOf(query.ToUpper()) != -1)
                    {
                        DisplayPrisoner(prisoner, index);
                    }
                    index++;
                }

            }
        }
        private void deleteMenuItem_Click(object sender, EventArgs e)
        {
            int index =
Convert.ToInt32(tablePrison.SelectedCells[0].OwningRow.Cells[0].Value);
            jailed.RemoveAt(index);
            DisplayList();
        }
        private void cellContextMenu_Opening(object sender, CancelEventArgs
e) // меню ячейки
        {
            Point selected = new Point(cellContextMenu.Left,
cellContextMenu.Top);
            selected = tablePrison.PointToClient(Cursor.Position);
            DataGridView.HitTestInfo selectedCell =
tablePrison.HitTest(selected.X, selected.Y);
            if (selectedCell.Type == DataGridViewHitTestType.Cell)
            {
                tablePrison.ClearSelection();
                tablePrison[selectedCell.ColumnIndex,
selectedCell.RowIndex].Selected = true;

            }
            else
            {
                e.Cancel = true;
            }
        }
        private void add_Click(object sender, EventArgs e)
        {
            Add_Form add = new Add_Form(jailed);
            add.Show();
        }

        private void editMenuItem_Click(object sender, EventArgs e)
//редактирование
        {
            int index =
Convert.ToInt32(tablePrison.SelectedCells[0].OwningRow.Cells[0].Value);
            Form editionForm = new Add_Form(index, jailed);
            editionForm.ShowDialog();
        }

        private void searchForm_Activated(object sender, EventArgs e)
        {
            DisplayList();
        }

        private void button1_Click(object sender, EventArgs e)
        {
            using (StreamWriter writer = new StreamWriter("Отчет о
формировании состава.txt", false, Encoding.Default))
            {
                for (int i = 0; i < tablePrison.RowCount; i++)
                {
                    writer.WriteLine("�мя : " + tablePrison[1,
i].Value.ToString());
                    writer.WriteLine("Фамилия : " + tablePrison[2,
i].Value.ToString());
                    writer.WriteLine("Камера : " + tablePrison[3,
i].Value.ToString());
                    writer.WriteLine("Статья : " + tablePrison[4,
i].Value.ToString());
                    writer.WriteLine("Дата заключения под стражу : " +
tablePrison[5, i].Value.ToString());
                    writer.WriteLine("Место в тюремной иерархии : " +
tablePrison[6, i].Value.ToString());
                    writer.WriteLine("Родстевеники : " + tablePrison[7,
i].Value.ToString());
                    writer.WriteLine("Характер : " + tablePrison[8,
i].Value.ToString());

                    writer.WriteLine();
                }
                writer.WriteLine("Количество заключеных : " +
tablePrison.RowCount);
                writer.Close();
            }
            MessageBox.Show("Выполнено");
            Process.Start("Отчет о формировании состава.txt");
        }

        private void searchTextBox_TextChanged(object sender, EventArgs e)
        {
            Search();
        }

        private void closeToolStripMenuItem_Click(object sender, EventArgs e)
        {
            Close();
        }

        private void infoToolStripMenuItem1_Click(object sender, EventArgs e)
        {
            Info info = new Info();
            info.ShowDialog();
        }

        private void OpenToolStripMenuItem_Click(object sender, EventArgs e)
        {
            IFormatter loader = new BinaryFormatter();
            OpenFileDialog dialog = new OpenFileDialog()
            {
                FileName = "DB.bin",
                Filter = "База данных (*.bin)|*.bin|Все файлы (*.*)|*.*"
            };
            if (dialog.ShowDialog() == DialogResult.OK)
            {
                Stream stream = new FileStream(dialog.FileName,
FileMode.Open, FileAccess.Read, FileShare.None);
                jailed = (Prison)loader.Deserialize(stream);
                stream.Close();
                Search();
            }
        }

        private void SaveToolStripMenuItem_Click(object sender, EventArgs e)
        {
            IFormatter saver = new BinaryFormatter();
            SaveFileDialog dialog = new SaveFileDialog()
            {
                FileName = "DB.bin",
                Filter = "База данных (*.bin)|*.bin|Все файлы (*.*)|*.*"
            };
            if (dialog.ShowDialog() == DialogResult.OK)
            {
                Stream stream = new FileStream(dialog.FileName,
FileMode.Create, FileAccess.Write, FileShare.None);
                saver.Serialize(stream, jailed);
                stream.Close();
            }
        }

        private void WorkersButton_Click(object sender, EventArgs e)
        {
            WorkersForm son = new WorkersForm(jailed.Workers);
            if (son.ShowDialog() == DialogResult.OK)
            {
                jailed.Workers = son.toSave;
            }
        }

        private void статистикаToolStripMenuItem_Click(object sender,
EventArgs e)
        {
            using (StreamWriter writer = new StreamWriter("Stats.txt", false,
Encoding.Default))
            {
                Dictionary<int,int> dict = new Dictionary<int, int>();
                foreach (Prisoner prisoner in jailed)
                {
                    if (!dict.ContainsKey(prisoner.Room))
dict.Add(prisoner.Room, 1);
                    else dict[prisoner.Room]++;
                }
                writer.WriteLine("Отчет по камерам");
                foreach (KeyValuePair<int,int> key in dict)
                {
                    writer.WriteLine("Камера №" + key.Key + ": " +
key.Value);
                }
                writer.WriteLine("Количество заключеных : " + jailed.Count);
                Dictionary<string, int> dict2 = new Dictionary<string,
int>();

                writer.WriteLine("Отчет по должностям рабочих:");
                foreach (Worker wrkr in jailed.Workers)
                {
                    if (!dict2.ContainsKey(wrkr.Position))
dict2.Add(wrkr.Position, 1);
                    else dict2[wrkr.Position]++;
                }
                foreach (KeyValuePair<string, int> key in dict2)
                {
                    writer.WriteLine(key.Key + ": " + key.Value);
                }
                writer.Close();
            }
            MessageBox.Show("Выполнено");
            Process.Start("Stats.txt");
        }
    }


    public partial class LoginForm : Form
    {
        //список аккаунтов
        public AccountList RC;
        public LoginForm()
        {
            InitializeComponent();
        }

        //Проверка полей и открытие главной формы
        private void LoginButton_Click(object sender, EventArgs e)
        {
            if(RC.ContainsByText(LoginBox.Text, PasswordBox.Text))
                OpenMainForm();
            else
            {
                MessageBox.Show("Такого пользователя не существует, либо
пароль не верен");
            }
        }

        //чтение из файла с аккаунтами
        private void LoginForm_Load(object sender, EventArgs e)
        {
            RC = AccountList.ReadFromFile();
        }

        //метод открытие главной формы
        public void OpenMainForm()
        {
            searchForm Main = new searchForm();
            Hide();
            Main.ShowDialog();
            Close();
        }

        //открытие формы редактирования списка аккаунтов
        private void ToAdminControl_Click(object sender, EventArgs e)
        {
            if (RC.ContainsByText(LoginBox.Text, PasswordBox.Text, true))
            {
                AdminLoginForm admin = new AdminLoginForm(RC);
                Hide();
                admin.ShowDialog();
                Show();
            }
            else
            {
                MessageBox.Show("У пользователя нет прав администратора или 
пользователя не существует");
            }
        }
    }


    public partial class AdminLoginForm : Form
    {
        //список аккаунтов
        public AccountList Basic;
        //конструктор формы, аргументом которого является редактируемый 
список
        public AdminLoginForm(AccountList RC)
        {
            InitializeComponent();
            Basic = RC;
            table.Columns.Add("ID");
            table.Columns.Add("Логин");
            table.Columns.Add("Пароль");
            table.Columns.Add("Права админа");
            DisplayList();
        }
        //таблица
        DataTable table = new DataTable();
        private void AdminLoginForm_Load(object sender, EventArgs e)
        {
        }

        //метод отображения списка
        private void DisplayList()
        {
            table.Rows.Clear();
            int index = 0;


            foreach (Account account in Basic)
            {
                DisplayAccount(account, index);
                index++;
            }
            AccView.DataSource = table;
            AccView.Columns[0].Visible = false;
        }

        //метод отображения аккаунта из коллекции
        private void DisplayAccount(Account account, int index)
        {
            DataRow row = table.NewRow();
            row["ID"] = index;
            row["Логин"] = account.Login;
            row["Пароль"] = account.Password;
            row["Права админа"] = account.IsAdmin;
            table.Rows.Add(row);
        }

        //метод удаления аккаунта из списка
        private void DeleteButton_Click(object sender, EventArgs e)
        {
            int index = 
Convert.ToInt32(AccView.SelectedCells[0].OwningRow.Cells[0].Value);
            Basic.RemoveAt(index);
            DisplayList();
        }
        //сохранение в файл и закрытие формы
        private void AdminLoginForm_FormClosing(object sender,
FormClosingEventArgs e)
        {
            Basic.SaveToFile();
        }

        private void CloseButton_Click(object sender, EventArgs e)
        {
            Close();
        }

        //кнопка добавления
        private void AddButton_Click(object sender, EventArgs e)
        {
            AdminEdit edit = new AdminEdit(Basic);
            edit.ShowDialog(this);
            DisplayList();
        }

        //кнопка редактирования
        private void EditButton_Click(object sender, EventArgs e)
        {
            int index =
Convert.ToInt32(AccView.SelectedCells[0].OwningRow.Cells[0].Value);
            AdminEdit edit = new AdminEdit(Basic, index);
            edit.ShowDialog(this);
            DisplayList();
        }
    }


    public partial class AdminEdit : Form
    {
        //флаг редактирования\добавления
        private bool edit;
        //индекс редактируемого элемента
        private int Index;
        //список в который добавляется\редактируется элемент
        private AccountList List;

        //Конструктор формы с заменой аккаунта (редактирование)
        public AdminEdit(AccountList list,int index)
        {
            InitializeComponent();
            LoginBox.Text = list[index].Login;
            PasswordBox.Text = list[index].Password;
            if (list[index].IsAdmin) AdminCheckBox.Checked = true;
            else AdminCheckBox.Checked = false;
            Index = index;
            List = list;
            edit = true;
        }

        //конструктор формы добавления аккаунта
        public AdminEdit(AccountList list)
        {
            InitializeComponent();
            List = list;
            edit = false;
        }

        //принятие изменений (сохранение в основной список)
        private void AcceptButton_Click(object sender, EventArgs e)
        {
            if (LoginBox.Text == "" || PasswordBox.Text == "")
            {
                MessageBox.Show("Проверьте заполненность полей");
                return;
            }
            if (edit)
            {
                List[Index] = new Account(LoginBox.Text, PasswordBox.Text,
AdminCheckBox.Checked);
            }
            else
            {
                List.Add(new Account(LoginBox.Text, PasswordBox.Text,
AdminCheckBox.Checked));
            }
            Close();
        }

        private void CancelButton_Click(object sender, EventArgs e)
        {
            Close();
        }
    }


    public partial class Add_Form : Form
    {

        //конструктор формы для добавления заключенных
        public Add_Form(Prison jailed)
        {
            InitializeComponent();
            this.jailed = jailed;
            this.forEdit = false;
        }

        //проверка даты
        public bool Data()
        {
            if (DateTime.Now.Year > Convert.ToInt32(Year.Text))
            {
                return true;
            }
            else
            {
                if (DateTime.Now.Year == Convert.ToInt32(Year.Text))
                {
                    if (DateTime.Now.Month > Convert.ToInt32(Month.Text))
                    {
                        return true;
                    }
                    else
                    {
                        if (DateTime.Now.Month ==
Convert.ToInt32(Month.Text))
                        {
                            if (DateTime.Now.Day >=
Convert.ToInt32(Day.Text)) return true;
                            else return false;
                        }
                        else return false;
                    }
                }
                else return false;
            }
        }

        public Prison jailed = null;
        private bool forEdit = false;
        private int index = 0;

        //констркутор формы для добавления заключенного
        public Add_Form(int index, Prison jailed)
        {
            InitializeComponent();
            this.jailed =jailed;
            this.forEdit = true;
            this.index = index;
            Prisoner prisoner = jailed[index];
            First_Name.Text = prisoner.First_Name;
            Second_Name.Text =prisoner.Second_Name;
            Room.Text = Convert.ToString(prisoner.Room);
            Criminality.Text = prisoner.Criminality;
            Day.Text = Convert.ToString(prisoner.DateOfJailed.Day);
            Month.Text = Convert.ToString(prisoner.DateOfJailed.Month);
            Year.Text = Convert.ToString(prisoner.DateOfJailed.Year);
            PITPH.Text = prisoner.PITPH;
            Relatives.Text = prisoner.Relatives;
            Temper.Text = prisoner.Temper;
        }

        //проверка и добавление в список
        private void Add_button_Click(object sender, EventArgs e)
        {
            int day = Convert.ToInt32(Day.Text);
            int month = Convert.ToInt32(Month.Text);
            if (day > 31 || day < 1 || month > 12 || month < 1)
            {
                MessageBox.Show("Неправильно введена дата");
                return;
            }
                if (requiredCheck() && Year.Text.Length == 4 && Data())
            {
                if (forEdit == false)
                {
                    jailed.Add(new Prisoner(First_Name.Text,
Second_Name.Text, Convert.ToInt32(Room.Text), Criminality.Text, new
Date(Convert.ToInt32(Day.Text), Convert.ToInt32(Month.Text),
Convert.ToInt32(Year.Text)), PITPH.Text, Relatives.Text, Temper.Text));

                    this.Close();
                }
                else
                {
                    jailed[index] = (new Prisoner(First_Name.Text,
Second_Name.Text, Convert.ToInt32(Room.Text), Criminality.Text, new
Date(Convert.ToInt32(Day.Text), Convert.ToInt32(Month.Text),
Convert.ToInt32(Year.Text)), PITPH.Text, Relatives.Text, Temper.Text));
                    this.Close();
                }
            }
         }
        private void cancelButton_Click(object sender, EventArgs e)
        {

            this.Close();
        }
        private void check(object sender, KeyPressEventArgs e) //поля только
с числами
        {
            if (!Char.IsDigit(e.KeyChar) && !Char.IsControl(e.KeyChar))
e.Handled = true;
        }

        private bool requiredCheck()  //проверка необходимых полей
        {
            bool result = checkOne(Room) & checkOne(Day) & checkOne(Month) &
checkOne(Year) & checkOne(First_Name) & checkOne(Second_Name) &
checkOne(Temper) ;
            return result;
        }

        private bool checkOne(TextBox input) //проверка одного поля
        {
            bool result = true;
            if (input.Text == "")
            {
                result = false;
                ToolTip req = new System.Windows.Forms.ToolTip();
                req.Show("Заполните это поле", input, 1000);
            }
            return result;
        }
    }


    //перименованный список работников
    [Serializable]
    public class WorkerCollection : List<Worker>
    {
    }


    //класс рабочего
    [Serializable]
    public class Worker
    {
        // �мя
        public string FirstName;
        // Фамилия
        public string SecondName;
        // Должность
        public string Position;
        // Дата наема
        public Date DateOfStart;

        //конструктор
        public Worker(string fName, string sName, string pos, Date
dateOfStart)
        {
            FirstName = fName;
            SecondName = sName;
            Position = pos;
            DateOfStart = dateOfStart;
        }
    }


    //класс заключенного
    [Serializable]
```

```
   public class Prisoner
    {
        //имя
        public string First_Name;
        //фамилия
        public string Second_Name;
        //камера
        public int Room;
        //
        public string Criminality;
        public Date DateOfJailed;
        //место в тюремной иерархии
        public string PITPH;
        //родественники
        public string Relatives;
        //характер
        public string Temper;

        public Prisoner()
        {
            this.First_Name = "";
            this.Second_Name = "";
            this.Room = 0;
            this.Criminality = "";
            this.PITPH = "";
            this.Relatives = "";
            this.Temper = "";
            this.DateOfJailed = new Date();


        }

        public Prisoner(string First_Name,string Second_Name,int Room, string
Criminality, Date DateOfJailed, string PITPH, string Relatives, string
Temper)
        {
            this.First_Name = First_Name;
            this.Second_Name = Second_Name;
            this.Room = Room;
            this.Criminality = Criminality;
            this.PITPH = PITPH;
            this.Relatives = Relatives;
            this.Temper = Temper;
            this.DateOfJailed = DateOfJailed;
        }

    }

    //класс даты
    [Serializable]
    public class Date
    {
        public int Day;
        public int Month;
        public int Year;

        public Date()
        {
        }

        public Date(int Day, int Month, int Year)
        {
            if(Day > 31 || Day < 1 || Month > 12 || Month < 1 || Year >
DateTime.Now.Year) throw new Exception();
```

```
            this.Day = Day;
            this.Month = Month;
            this.Year = Year;
        }


    }


    [Serializable]
   public class Prison : List<Prisoner>
    {
        public WorkerCollection Workers = new WorkerCollection();
    }


    [Serializable]
    public class AccountList : List<Account>
    {
        //метод чтения из файла
        public void SaveToFile()
        {
            BinaryFormatter formatter = new BinaryFormatter();
            FileStream fs = new FileStream("accounts.bin", FileMode.Create,
FileAccess.Write, FileShare.None);
            formatter.Serialize(fs,this);
            fs.Close();
        }

        //метод чтения из файла
        public static AccountList ReadFromFile()
        {
            BinaryFormatter formatter = new BinaryFormatter();
            FileStream fs = new FileStream("accounts.bin", FileMode.Open,
FileAccess.Read, FileShare.None);
            AccountList result = (AccountList) formatter.Deserialize(fs);
            fs.Close();
            return result;
        }

        //метод который возвращает наличие аккаунта в списке и проверка на
права администратора
        public bool ContainsByText(string login, string password, bool
adminCheck = false)
        {
            foreach (Account account in this)
            {
                if (account.Login == login && account.Password == password)
                {
                    if (adminCheck) return account.IsAdmin;
                    return true;
                }
            }
            return false;
        }

    }


    [Serializable]
    public class Account
    {
        public string Login;
        public string Password;
        public bool IsAdmin;

        public Account(string login, string pass, bool admin = false)
        {
            Login = login;
            Password = pass;
            IsAdmin = admin;
        }
    }
```