МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ   
ФЕДЕРАЦИИ

Федеральное государственное бюджетное образовательное учреждение высшего профессионального образования

«Вятский государственный университет»

Факультет автоматики и вычислительной техники

Кафедра электронных вычислительных машин

Исследование алгоритмов сортировки данных на внешней памяти.

Отчет

Лабораторная работа №8 по дисциплине

«Программирование»

Выполнил студент группы ИВТ-11 \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_/Коротаев/

Проверил преподаватель\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_/Чистяков Г.А./

Киров 2020

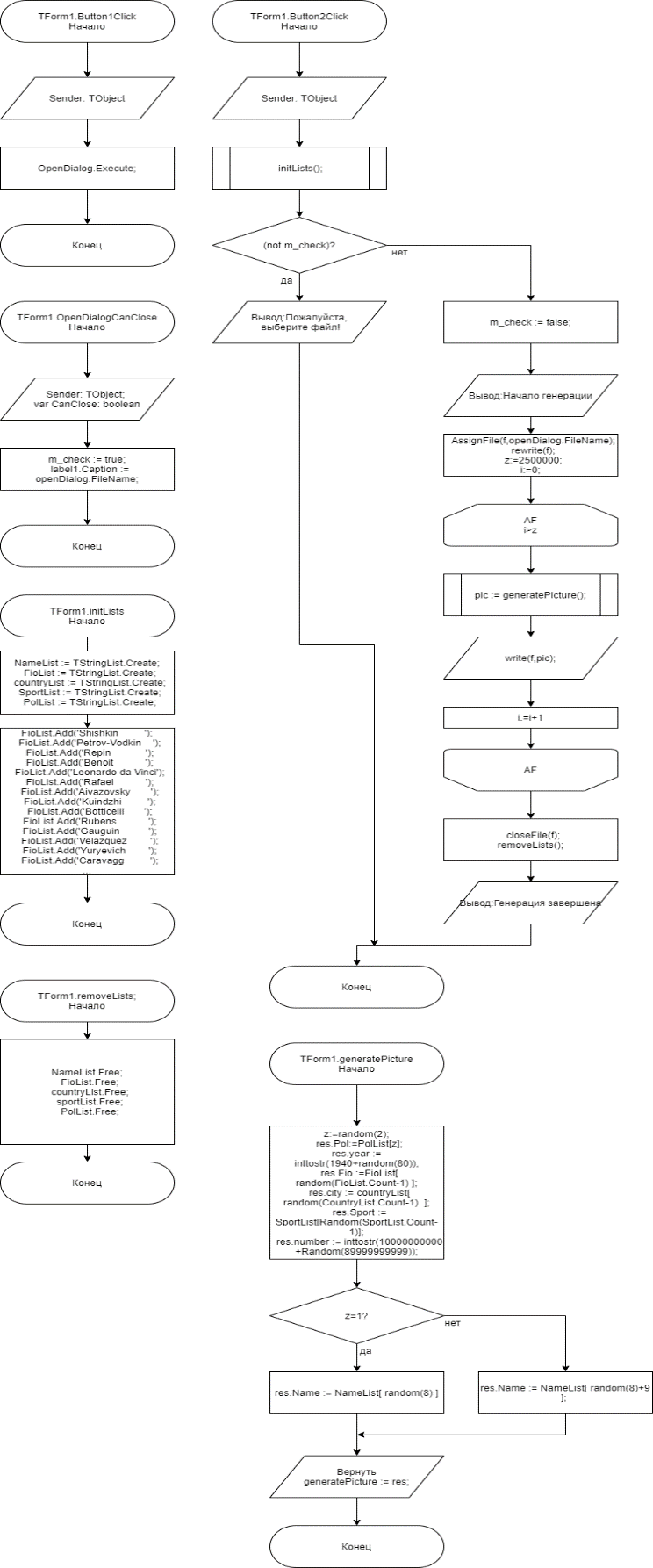
Вариант:0

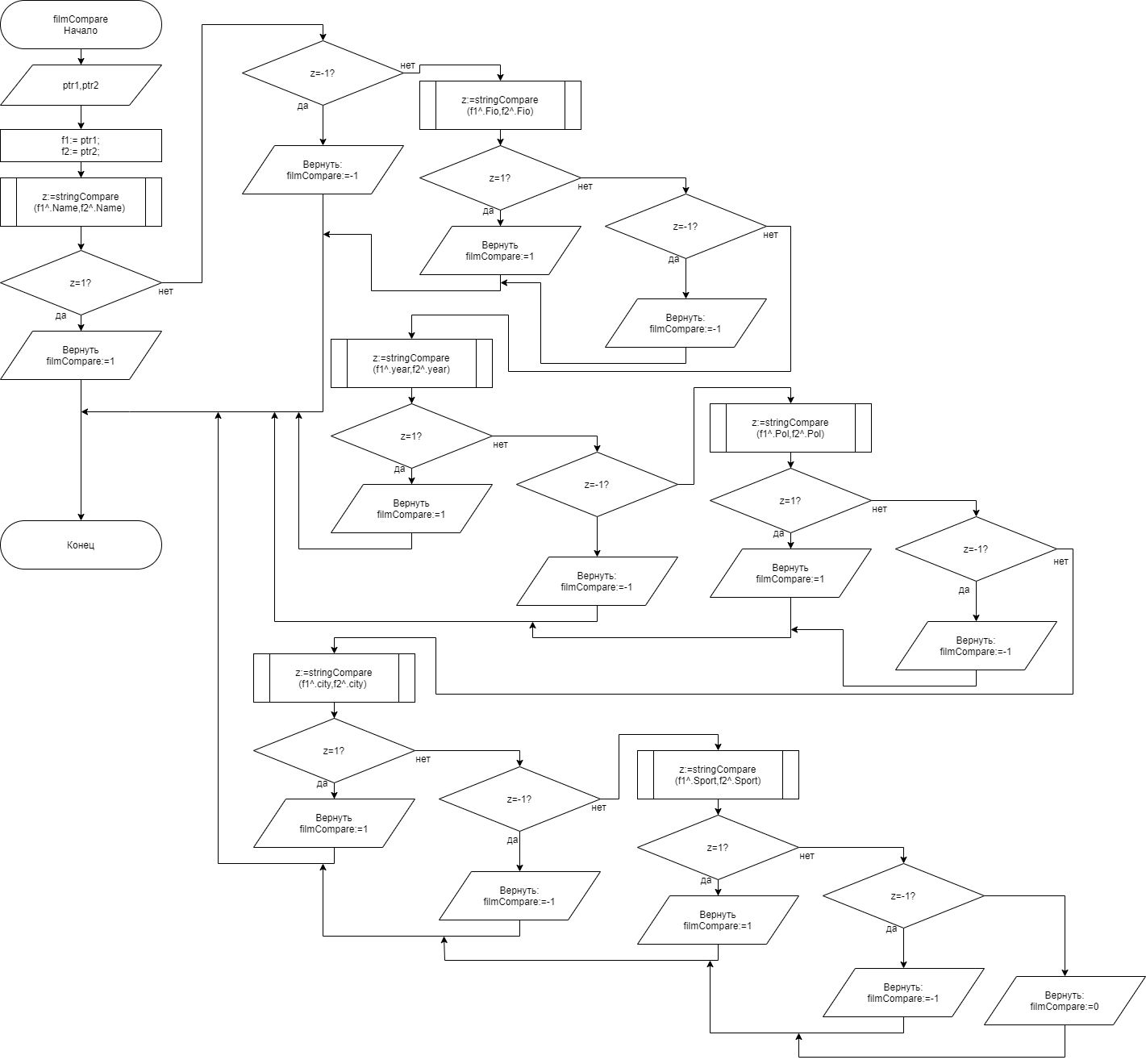
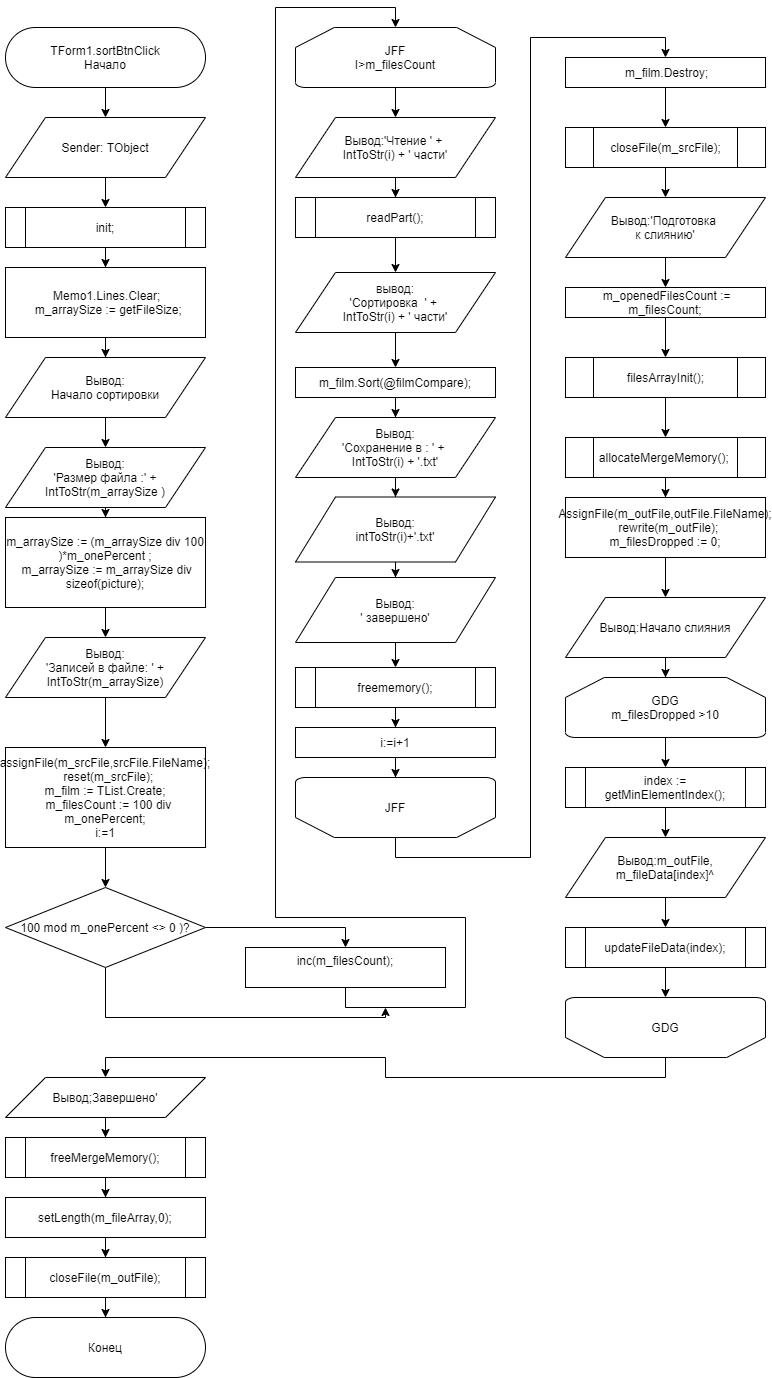
Цель работы: познакомиться с организацией стандартных диалоговых окон, изучить принципы работы с данными на внешней памяти, получить навыки работы с типизированными файлами.

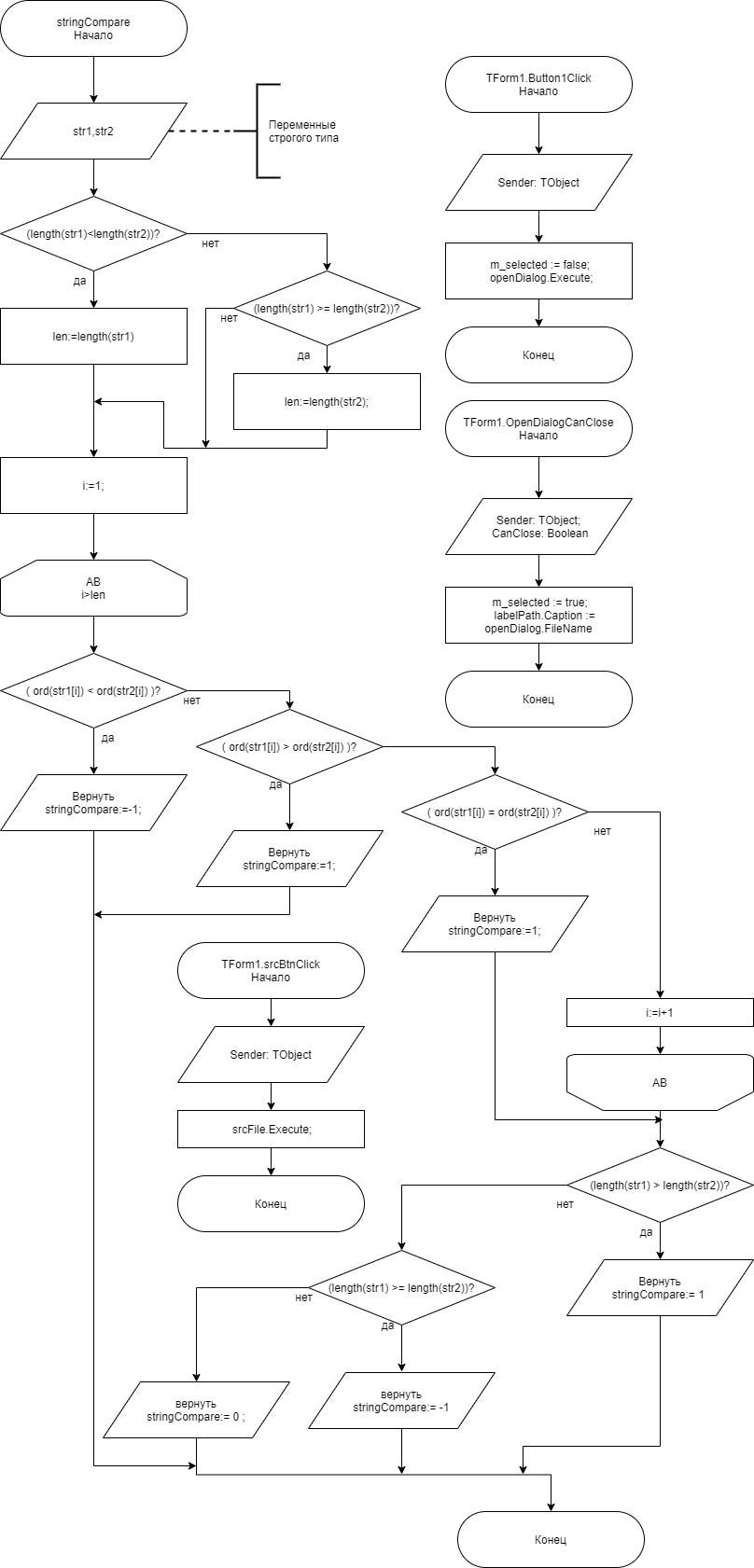
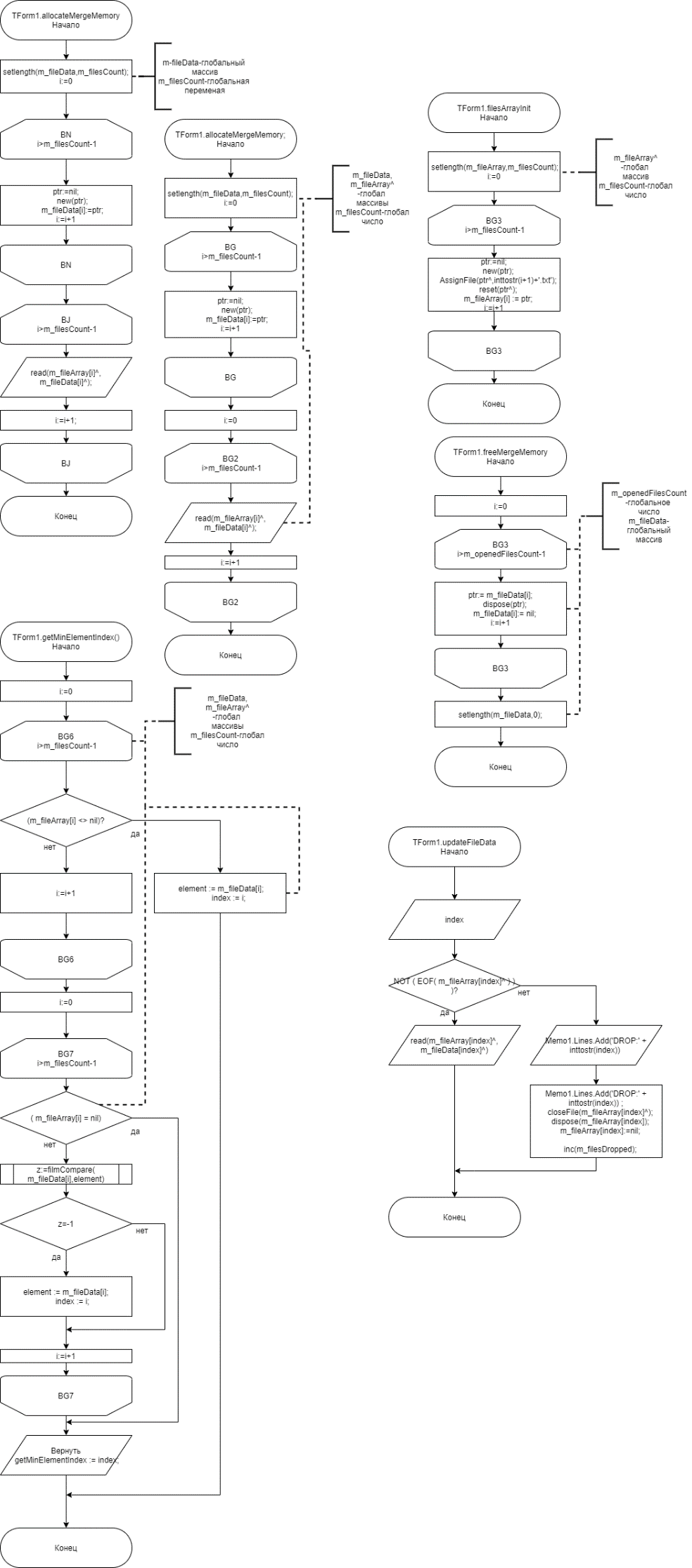
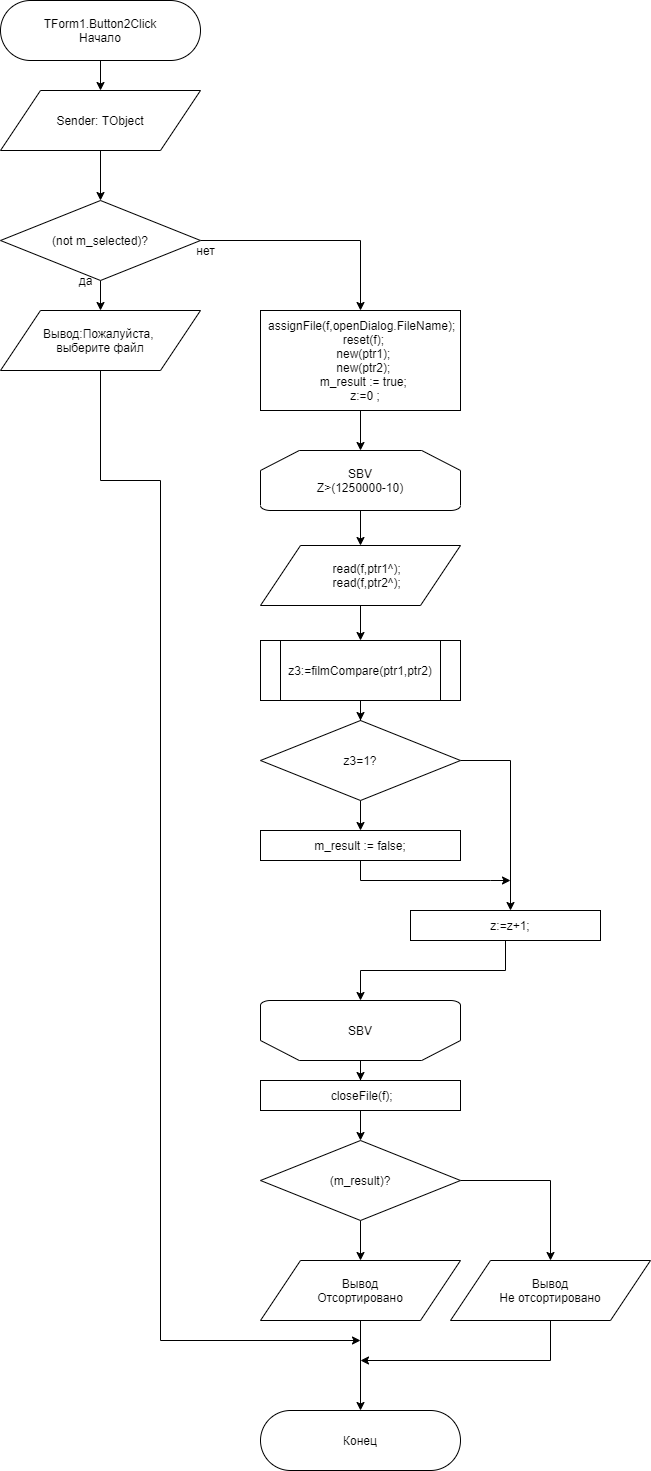
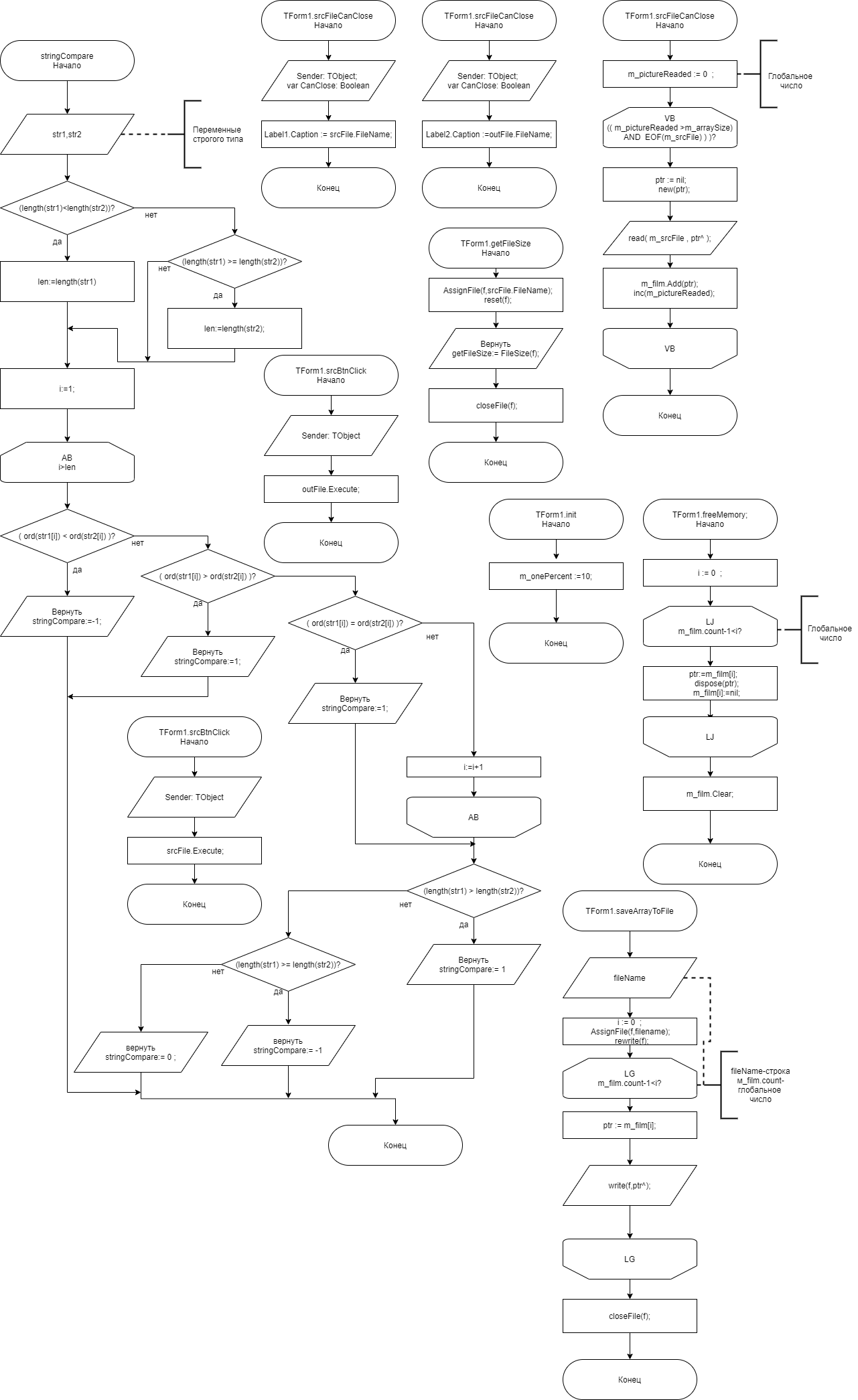
Задание:

1. Разработать программу для сортировки данных, находящихся на жестком диске в типизированном файле.  
2. Размер соритруемых данных должен быть не менее 1 Гб. Объем потребляемой оперативной памяти не должен превышать 10% от размера исходного файла.  
3. Исходный и целевой файлы должны выбираться с помощью стандартных диалоговых окон.  
4. Структуру сортируемого файла и ключ сортировки согласовать с преподавателем.  
5. Помимо основной программы, требуется реализовать две вспомогательные утилиты - для генерации массива данных и проверки упорядоченности.

Схема алгоритма







Исходный код

1. Генератор

**unit** Unit1;

{$mode objfpc}{$H+}

**interface**

**uses**

SysUtils, Variants, Classes, Graphics, Controls, Forms,

Dialogs, StdCtrls, MaskEdit;

**type** picture=**record**

Name: string[100];

Fio: string[100];

Pol: string[5];

year : string[4];

city : string[100];

Sport: string[100];

number :string[11];

**end**;

**type**

{ TForm1 }

TForm1 = **class**(TForm)

Button1: TButton;

Button2: TButton;

Label1: TLabel;

Label2: TLabel;

Memo1: TMemo;

OpenDialog: TOpenDialog;

**procedure** Button1Click(Sender: TObject);

**procedure** Button2Click(Sender: TObject);

**procedure** OpenDialogCanClose(Sender: TObject; **var** CanClose: boolean);

**function** generatePicture: picture;

**procedure** initLists();

**procedure** removeLists();

**private**

{ Private declarations }

**public**

{ Public declarations }

**end**;

**var**

Form1: TForm1;

m\_check : boolean;

NameList,FioList,countryList,sportList ,PolList: TStringList;

z:integer;

**implementation**

{$R \*.lfm}

**procedure** TForm1.Button1Click(Sender: TObject);

**begin**

OpenDialog.Execute;

**end**;

**procedure** TForm1.Button2Click(Sender: TObject);

**var** f : **file of** picture;

i,z: longint;

pic : picture;

**begin**

initLists();

**if**(**not** m\_check) **then**

**begin**

Memo1.Lines.Add('Пожалуйста, выберите файл!');

**exit**;

**end**;

m\_check := false;

Memo1.Lines.Add('Начало генерации');

AssignFile(f,openDialog.FileName);

rewrite(f);

z:=2500000;

**for** i:=0 **to** z **do**

**begin**

pic := generatePicture();

write(f,pic);

**end**;

closeFile(f);

Memo1.Lines.Add('Генерация завершена');

removeLists();

**end**;

**procedure** TForm1.OpenDialogCanClose(Sender: TObject; **var** CanClose: boolean);

**begin**

m\_check := true;

label1.Caption := openDialog.FileName;

**end**;

**function** TForm1.generatePicture: picture;

**var** res: picture;

z:integer;

**begin**

z:=random(2);

res.Pol:=PolList[z];

**case** z **of**

1: **begin** res.Name := NameList[ random(8) ]; **end**;

0:**begin** res.Name := NameList[ random(8)+9 ]; **end**;

**end**;

res.year := inttostr(1940+random(80));

res.Fio :=FioList[ random(FioList.Count-1) ];

res.city := countryList[ random(CountryList.Count-1) ];

res.Sport := SportList[Random(SportList.Count-1)];

res.number := inttostr(10000000000+Random(89999999999));

generatePicture := res;

**end**;

**procedure** TForm1.initLists();

**begin**

NameList := TStringList.Create;

FioList := TStringList.Create;

countryList := TStringList.Create;

SportList := TStringList.Create;

PolList := TStringList.Create;

//Пол

PolList.Add('Women');

PolList.Add('Men ');

//Мужсике Фамилии 0-27

FioList.Add('Shishkin ');

FioList.Add('Petrov-Vodkin ');

FioList.Add('Repin ');

FioList.Add('Benoit ');

FioList.Add('Leonardo da Vinci');

FioList.Add('Rafael ');

FioList.Add('Aivazovsky ');

FioList.Add('Kuindzhi ');

FioList.Add('Botticelli ');

FioList.Add('Rubens ');

FioList.Add('Gauguin ');

FioList.Add('Velazquez ');

FioList.Add('Yuryevich ');

FioList.Add('Caravagg ');

FioList.Add('Aleksandrovich ');

FioList.Add('Levitan ');

FioList.Add('Malevich ');

FioList.Add('Degas ');

FioList.Add('Brullov ');

FioList.Add('Goya ');

FioList.Add('Delacroix ');

FioList.Add('Buonarotti ');

FioList.Add('Durer ');

FioList.Add('Bilibin ');

FioList.Add('Vrubel ');

FioList.Add('Kustodiev ');

FioList.Add('Rylov ');

FioList.Add('Serebryakova ');

//Мужские имена 0-8

NameList.Add('Anatoliy ');

NameList.Add('Anton ');

NameList.Add('Vladimir ');

NameList.Add('Igor ');

NameList.Add('Dmitriy ');

NameList.Add('Oleg ');

NameList.Add('Sergey ');

NameList.Add('Fedor ');

NameList.Add('Kirill ');

//Женские Имена 9-17

NameList.Add('Anzhela ');

NameList.Add('Galina ');

NameList.Add('Inna ');

NameList.Add('Kristina ');

NameList.Add('Oksana ');

NameList.Add('Regina ');

NameList.Add('Snezhana ');

NameList.Add('Larisa ');

NameList.Add('Kira ');

countryList.Add('Kirov ');

countryList.Add('Moscow ');

countryList.Add('Saint-Petersburg');

countryList.Add('Yekaterinburg ');

countryList.Add('Nizhny Novgorod ');

countryList.Add('Kazan ');

countryList.Add('Paris ');

countryList.Add('London ');

countryList.Add('Pavlovsk ');

sportList.Add('baseball ');

sportList.Add('basketball ');

sportList.Add('football ');

sportList.Add('volleybal ');

sportList.Add('cricket ');

sportList.Add('dog racing ');

sportList.Add('horse racing ');

sportList.Add('horseback riding ');

sportList.Add('rowing ');

sportList.Add('cross country race ');

sportList.Add('waterskiing ');

sportList.Add('alpine skiing ');

sportList.Add('ice hockey ');

sportList.Add('BASE jumping ');

sportList.Add('rock climbing ');

sportList.Add('skysurfing ');

sportList.Add('bowling ');

sportList.Add('draught ');

sportList.Add('car racing ');

sportList.Add('handball ');

sportList.Add('mountaineering ');

sportList.Add('weightlifting ');

**end**;

**procedure** TForm1.removeLists;

**begin**

NameList.Free;

FioList.Free;

countryList.Free;

sportList.Free;

PolList.Free;

**end**;

**end**.

1. Сортировка

**unit** Unit1;

{$mode objfpc}{$H+}

**interface**

**uses**

SysUtils, Variants, Classes, Graphics, Controls, Forms,

Dialogs, StdCtrls;

**type** picture=**record**

Name: string[100];

Fio: string[100];

Pol: string[5];

year : string[4];

city : string[100];

Sport: string[100];

number :string[11];

**end**;

**type** link = ^picture;

cmpFunc = **function** (f1,f2:link):boolean;

pictureFile = **file of** picture;

pictureFileLink = ^pictureFile;

{ TForm1 }

TForm1 = **class**(TForm)

Label1: TLabel;

OpenDialog1: TOpenDialog;

OpenDialog2: TOpenDialog;

srcBtn: TButton;

Label2: TLabel;

outBtn: TButton;

srcFile: TOpenDialog;

outFile: TOpenDialog;

sortBtn: TButton;

Memo1: TMemo;

**procedure** srcBtnClick(Sender: TObject);

**procedure** outBtnClick(Sender: TObject);

**procedure** srcFileCanClose(Sender: TObject; **var** CanClose: Boolean);

**procedure** outFileCanClose(Sender: TObject; **var** CanClose: Boolean);

**procedure** sortBtnClick(Sender: TObject);

**procedure** init();

// function filmCompare(ptr1,ptr2:pointer):integer;

**function** getFileSize():integer;

**procedure** readPart();

**procedure** freeMemory();

**procedure** saveArrayToFile(fileName:string);

**procedure** filesArrayInit();

**procedure** allocateMergeMemory();

**procedure** freeMergeMemory();

**procedure** updateFileData(index : integer );

**function** getMinElementIndex():integer;

**private**

**public**

m\_film : TList;

m\_fileArray : **array of** pictureFileLink;

m\_fileData : **array of** link;

m\_arraySize : integer;

m\_pictureReaded : integer;

m\_filesCount : integer;

m\_onePercent : integer;

m\_openedFilesCount : integer;

m\_srcFile : **file of** picture;

m\_outFile : **file of** picture;

m\_filesDropped : integer;

**end**;

**var**

Form1: TForm1;

**implementation**

{$R \*.lfm}

**function** stringCompare(str1,str2:string):integer;

**var** i: integer;

len : integer;

**begin**

**if**(length(str1)<length(str2))**then**

len:=length(str1)

**else**

**if**(length(str1) >= length(str2))

**then**

len:=length(str2);

**for** i:=1 **to** len **do**

**begin**

// ord of empty string = 0

**if**( ord(str1[i]) < ord(str2[i]) )

**then begin**

stringCompare:=-1;

**exit**;

**end else**

**if**( ord(str1[i]) > ord(str2[i]) )

**then begin**

stringCompare:=1;

**exit**;

**end else**

**if**( ord(str1[i]) = ord(str2[i]) )

**then begin**

**continue**;

**end**;

**end**;

**if**(length(str1) > length(str2)) **then**

stringCompare:= 1 **else**

**if**(length(str1) < length(str2)) **then**

stringCompare:= -1 **else**

stringCompare:= 0 ;

**end**;

**function** filmCompare(ptr1,ptr2:pointer):integer;

**var** f1,f2 : link;

**begin**

f1:= ptr1;

f2:= ptr2;

**if**( stringCompare(f1^.Name,f2^.Name) = 1 ) **then**

filmCompare:=1 **else**

**if**( stringCompare(f1^.Name,f2^.Name) = -1 ) **then**

filmCompare:=-1 **else**

**if**( stringCompare(f1^.Fio,f2^.Fio) = 1 ) **then**

filmCompare:=1 **else**

**if**( stringCompare(f1^.Fio,f2^.Fio) = -1 ) **then**

filmCompare:=-1 **else**

**if**( stringCompare(f1^.year,f2^.year) = 1 ) **then**

filmCompare:=1 **else**

**if**( stringCompare(f1^.year,f2^.year) = -1 ) **then**

filmCompare:=-1 **else**

**if**( stringCompare(f1^.Pol,f2^.Pol) = 1 ) **then**

filmCompare:=1 **else**

**if**( stringCompare(f1^.Pol,f2^.Pol) = -1 ) **then**

filmCompare:=-1

**else**

**if**( stringCompare(f1^.city,f2^.city) = 1 ) **then**

filmCompare:=1 **else**

**if**( stringCompare(f1^.city,f2^.city) = -1 ) **then**

filmCompare:=-1 **else**

**if**( stringCompare(f1^.Sport,f2^.Sport) = 1 ) **then**

filmCompare:=1 **else**

**if**( stringCompare(f1^.Sport,f2^.Sport) = -1 ) **then**

filmCompare:=-1 **else**

filmCompare := 0;

**end**;

//opening a file with records

**procedure** TForm1.srcBtnClick(Sender: TObject);

**begin**

srcFile.Execute;

**end**;

//opening a file for output

**procedure** TForm1.outBtnClick(Sender: TObject);

**begin**

outFile.Execute;

**end**;

//closing the dirictory

**procedure** TForm1.srcFileCanClose(Sender: TObject; **var** CanClose: Boolean);

**begin**

Label1.Caption := srcFile.FileName;

**end**;

//closing the dirictory

**procedure** TForm1.outFileCanClose(Sender: TObject; **var** CanClose: Boolean);

**begin**

Label2.Caption :=outFile.FileName;

**end**;

// main sort function

**procedure** TForm1.sortBtnClick(Sender: TObject);

**var** filesCount,i,index : integer ;

**begin**

init;

// get file info,size of part and other...

Memo1.Lines.Clear;

Memo1.Lines.Add('Начало сортировки');

m\_arraySize := getFileSize;

Memo1.Lines.Add('Размер файла :' + IntToStr(m\_arraySize ));

m\_arraySize := (m\_arraySize **div** 100 )\*m\_onePercent ;

m\_arraySize := m\_arraySize **div sizeof**(picture);

Memo1.Lines.Add('Записей в файле: ' + IntToStr(m\_arraySize));

// opening source file

assignFile(m\_srcFile,srcFile.FileName);

reset(m\_srcFile);

// create list

m\_film := TList.Create;

m\_filesCount := 100 **div** m\_onePercent;

**if**(100 **mod** m\_onePercent <> 0 ) **then** inc(m\_filesCount);

// divide file on chunks, all chunks reading,sorting and writing to temp file

**for** i := 1 **to** m\_filesCount **do**

**begin**

Memo1.Lines.Add('Чтение ' + IntToStr(i) + ' части');

readPart();

Memo1.Lines.Add('Сортировка ' + IntToStr(i) + ' части');

m\_film.Sort(@filmCompare);

Memo1.Lines.Add('Сохранение в : ' + IntToStr(i) + '.txt');

saveArrayToFile(intToStr(i)+'.txt');

Memo1.Lines.Add(' завершено' ) ;

Memo1.Lines.Add('');

// list clearing

freememory();

**end**;

m\_film.Destroy;

closeFile(m\_srcFile);

// merge time

Memo1.Lines.Add('Подготовка к слиянию');

m\_openedFilesCount := m\_filesCount;

// opening all files

filesArrayInit();

// create array for values,and read first value

allocateMergeMemory();

// opening output file

AssignFile(m\_outFile,outFile.FileName);

rewrite(m\_outFile);

Memo1.Lines.Add('Начало слияния') ;

m\_filesDropped := 0;

**while**(m\_filesDropped <10) **do**

**begin**

// find lowest element in array of values

index := getMinElementIndex();

// write value to file

write(m\_outFile,m\_fileData[index]^);

// read next value from this file

updateFileData(index);

**end**;

Memo1.Lines.Add('Завершено');

// dispose filedata memory

freeMergeMemory();

// destroy m\_fileArray ( after disposing in merge )

setLength(m\_fileArray,0);

closeFile(m\_outFile);

**end**;

**function** TForm1.getFileSize: integer;

**var** f : **file of** byte;

**begin**

AssignFile(f,srcFile.FileName);

reset(f);

getFileSize:= FileSize(f);

closeFile(f);

**end**;

**procedure** TForm1.init;

**begin**

m\_onePercent :=10;

**end**;

**procedure** TForm1.readPart();

**var** ptr:link;

**begin**

m\_pictureReaded := 0 ;

**while**( ( m\_pictureReaded < m\_arraySize) **AND** (**NOT** EOF(m\_srcFile) ) ) **do**

**begin**

ptr := nil;

**new**(ptr);

read( m\_srcFile , ptr^ );

m\_film.Add(ptr);

inc(m\_pictureReaded);

**end**;

**end**;

**procedure** TForm1.freeMemory;

**var** i : integer;

ptr : link ;

**begin**

**for** i:= 0 **to** m\_film.Count-1 **do**

**begin**

ptr:=m\_film[i];

dispose(ptr);

m\_film[i]:=nil;

**end**;

m\_film.Clear;

**end**; ///

**procedure** TForm1.saveArrayToFile(fileName:string);

**var** f : **file of** picture;

ptr : link;

i:integer;

**begin**

AssignFile(f,filename);

rewrite(f);

**for** i:= 0 **to** m\_film.Count-1 **do**

**begin**

ptr := m\_film[i];

write(f,ptr^);

**end**;

closeFile(f);

**end**;

**procedure** TForm1.allocateMergeMemory;

**var** i:integer;

ptr : link;

**begin**

setlength(m\_fileData,m\_filesCount);

// allocate memory for all elements

**for** i:=0 **to** m\_filesCount-1 **do**

**begin**

ptr:=nil;

**new**(ptr);

m\_fileData[i]:=ptr;

**end**;

// write first elements..

**for** i:=0 **to** m\_filesCount-1 **do**

**begin**

read(m\_fileArray[i]^,m\_fileData[i]^);

**end**;

**end**;

**procedure** TForm1.filesArrayInit;

**var** i:integer;

ptr : pictureFileLink;

**begin**

setlength(m\_fileArray,m\_filesCount);

**for** i:= 0 **to** m\_filesCount-1 **do**

**begin**

ptr:=nil;

**new**(ptr);

AssignFile(ptr^,inttostr(i+1)+'.txt');

reset(ptr^);

m\_fileArray[i] := ptr;

**end**;

**end**;

**procedure** TForm1.freeMergeMemory;

**var** ptr : link;

i:integer;

**begin**

**for** i:= 0 **to** m\_openedFilesCount-1 **do**

**begin**

ptr:= m\_fileData[i];

dispose(ptr);

m\_fileData[i]:= nil;

**end**;

setlength(m\_fileData,0);

**end**;

**function** TForm1.getMinElementIndex():integer;

**var** i,index:integer;

element : link;

**begin**

**for** i:=0 **to** m\_filesCount-1 **do**

**begin**

**if**(m\_fileArray[i] <> nil)**then**

**begin**

element := m\_fileData[i];

index := i;

**break**;

**end**;

**end**;

**for** i:=0 **to** m\_filesCount-1 **do**

**begin**

**if** ( m\_fileArray[i] = nil) **then continue**;

**if** ( filmCompare( m\_fileData[i],element) = -1) **then**

**begin**

element := m\_fileData[i];

index := i;

**end**;

**end**;

getMinElementIndex := index;

**end**;

**procedure** TForm1.updateFileData(index: integer);

**var** i : integer;

swap : link;

**begin**

**if**( **NOT** ( EOF( m\_fileArray[index]^ ) ) ) **then**

read(m\_fileArray[index]^,m\_fileData[index]^)

**else begin**

Memo1.Lines.Add('DROP:' + inttostr(index)) ;

closeFile(m\_fileArray[index]^);

dispose(m\_fileArray[index]);

m\_fileArray[index]:=nil;

inc(m\_filesDropped);

**end**;

**end**;

**end**.

1. Чекер

**unit** Unit1;

{$mode objfpc}{$H+}

**interface**

**uses**

Windows, Messages, SysUtils, Variants, Classes, Graphics, Controls, Forms,

Dialogs, StdCtrls;

**type** picture=**record**

Name: string[100];

Fio: string[100];

Pol: string[5];

year : string[4];

city : string[100];

Sport: string[100];

number :string[11];

**end**;

link = ^picture;

**type**

{ TForm1 }

TForm1 = **class**(TForm)

Button1: TButton;

Label1: TLabel;

Label2: TLabel;

labelResult: TLabel;

labelPath: TLabel;

Button2: TButton;

OpenDialog: TOpenDialog;

**procedure** Button1Click(Sender: TObject);

**procedure** OpenDialogCanClose(Sender: TObject; **var** CanClose: Boolean);

**procedure** Button2Click(Sender: TObject);

**private**

m\_selected,m\_result : boolean;

{ Private declarations }

**public**

{ Public declarations }

**end**;

**var**

Form1: TForm1;

**implementation**

{$R \*.lfm}

**function** stringCompare(str1,str2:string):integer;

**var** i: integer;

len : integer;

**begin**

**if**(length(str1)<length(str2))**then**

len:=length(str1)

**else**

**if**(length(str1) >= length(str2))

**then**

len:=length(str2);

**for** i:=1 **to** len **do**

**begin**

// ord of empty string = 0

**if**( ord(str1[i]) < ord(str2[i]) )

**then begin**

stringCompare:=-1;

**exit**;

**end else**

**if**( ord(str1[i]) > ord(str2[i]) )

**then begin**

stringCompare:=1;

**exit**;

**end else**

**if**( ord(str1[i]) = ord(str2[i]) )

**then begin**

**continue**;

**end**;

**end**;

**if**(length(str1) > length(str2)) **then**

stringCompare:= 1 **else**

**if**(length(str1) < length(str2)) **then**

stringCompare:= -1 **else**

stringCompare:= 0 ;

**end**;

**function** filmCompare(ptr1,ptr2:pointer):integer;

**var** f1,f2 : link;

**begin**

f1:= ptr1;

f2:= ptr2;

**if**( stringCompare(f1^.Name,f2^.Name) = 1 ) **then**

filmCompare:=1 **else**

**if**( stringCompare(f1^.Name,f2^.Name) = -1 ) **then**

filmCompare:=-1 **else**

**if**( stringCompare(f1^.Fio,f2^.Fio) = 1 ) **then**

filmCompare:=1 **else**

**if**( stringCompare(f1^.Fio,f2^.Fio) = -1 ) **then**

filmCompare:=-1 **else**

**if**( stringCompare(f1^.year,f2^.year) = 1 ) **then**

filmCompare:=1 **else**

**if**( stringCompare(f1^.year,f2^.year) = -1 ) **then**

filmCompare:=-1 **else**

**if**( stringCompare(f1^.Pol,f2^.Pol) = 1 ) **then**

filmCompare:=1 **else**

**if**( stringCompare(f1^.Pol,f2^.Pol) = -1 ) **then**

filmCompare:=-1

**else**

**if**( stringCompare(f1^.city,f2^.city) = 1 ) **then**

filmCompare:=1 **else**

**if**( stringCompare(f1^.city,f2^.city) = -1 ) **then**

filmCompare:=-1 **else**

**if**( stringCompare(f1^.Sport,f2^.Sport) = 1 ) **then**

filmCompare:=1 **else**

**if**( stringCompare(f1^.Sport,f2^.Sport) = -1 ) **then**

filmCompare:=-1 **else**

filmCompare := 0;

**end**;

**procedure** TForm1.Button1Click(Sender: TObject);

**begin**

m\_selected := false;

openDialog.Execute;

**end**;

**procedure** TForm1.OpenDialogCanClose(Sender: TObject;

**var** CanClose: Boolean);

**begin**

m\_selected := true;

labelPath.Caption := openDialog.FileName

**end**;

**procedure** TForm1.Button2Click(Sender: TObject);

**var** f : **file of** picture;

ptr1,ptr2 : link;

z:integer;

**begin**

**if**(**not** m\_selected) **then**

**begin**

labelResult.Caption := 'Пожалуйста, выберите файл';

**exit**;

**end**;

assignFile(f,openDialog.FileName);

reset(f);

**new**(ptr1);

**new**(ptr2);

m\_result := true;

z:=0 ;

**while** {( not eof(f) )} z<(1250000-10) **do**

**begin**

read(f,ptr1^);

read(f,ptr2^);

**if** ( filmCompare(ptr1,ptr2) = 1 )

**then** m\_result := false;

z:=z+1;

**end**;

closeFile(f);

**if**(m\_result) **then**

labelResult.Caption := 'Отсортировано' **else**

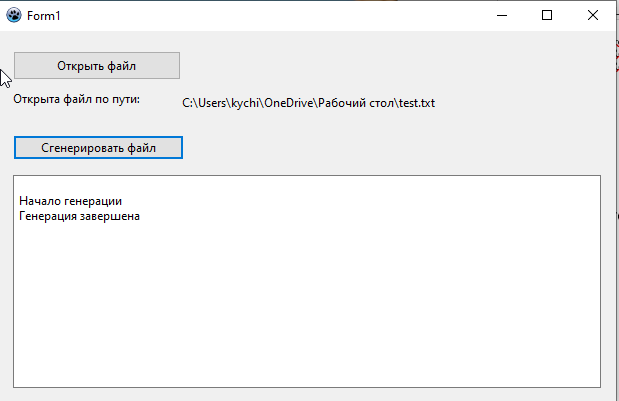
labelResult.Caption := 'Не отсортировано';

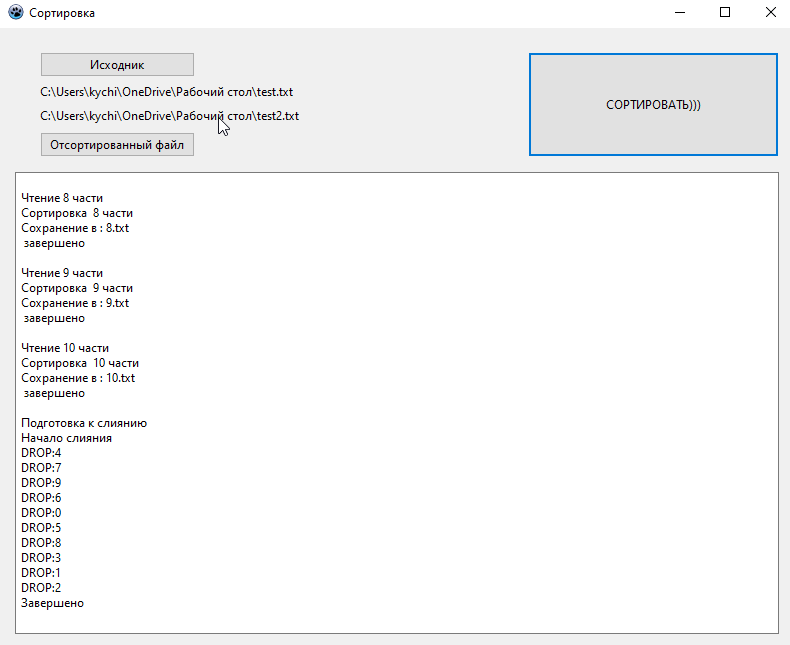
**end**;

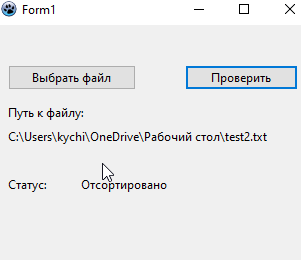
**end**.

**end**.

Фото отчет выполненой программы:







Вывод: В ходе выполнение лабораторной работы по программированию были получены:

* Навыки работы с типизированным файломю
* Навыки работы с лазарусом.
* Навык работы с сортировкой на внешней памяти .

При выполнение лабораторной работы были выявлены сложности такие как:

* Работа с сортировкой
* Работа с файлом.