МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ РОССИЙСКОЙ ФЕДЕРАЦИИ

Федеральное государственное бюджетное образовательное учреждение

Высшего профессионального образования

**«Вятский государственный университет»**

**(ФГБОУ ВПО «ВятГУ»)**

Факультет автоматики и вычислительной техники

Кафедра электронных вычислительных машин

Исследование алгоритмов сортировки данных на внешней памяти

Отчет по лабораторной работе №8 дисциплины

«Программирование»

Выполнил студент группы ИВТб-11\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_/Седов М.Д./

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

Киров 2018

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

**Задание:**

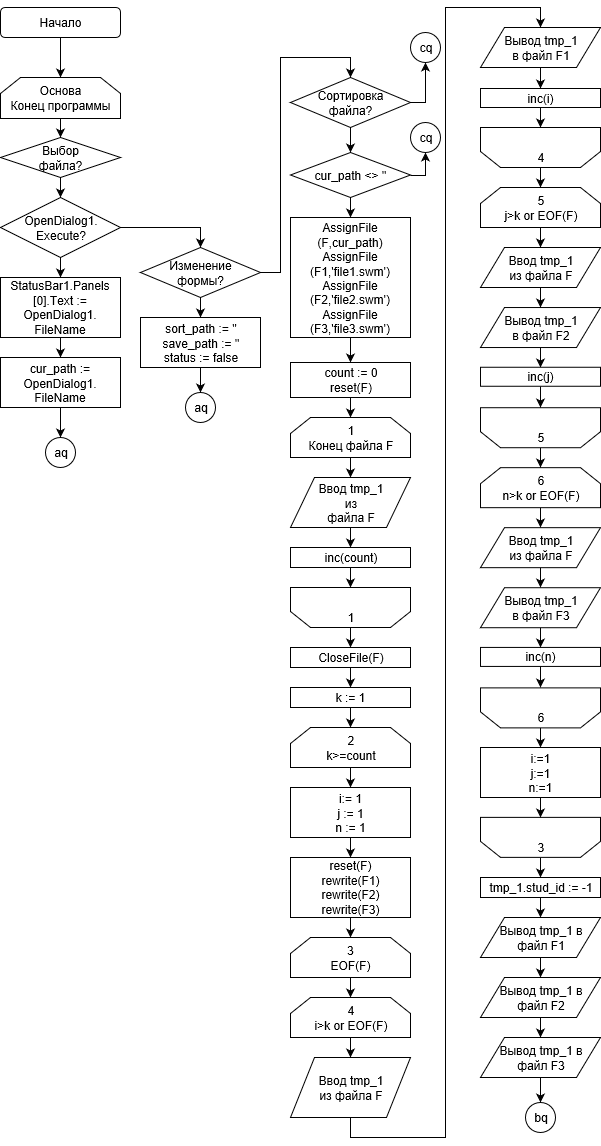
**1.Разработать программу для сортировки данных, находящихся на жестком диске в типизированном файле.**

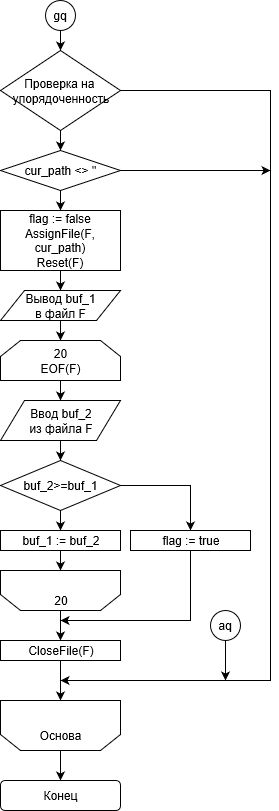
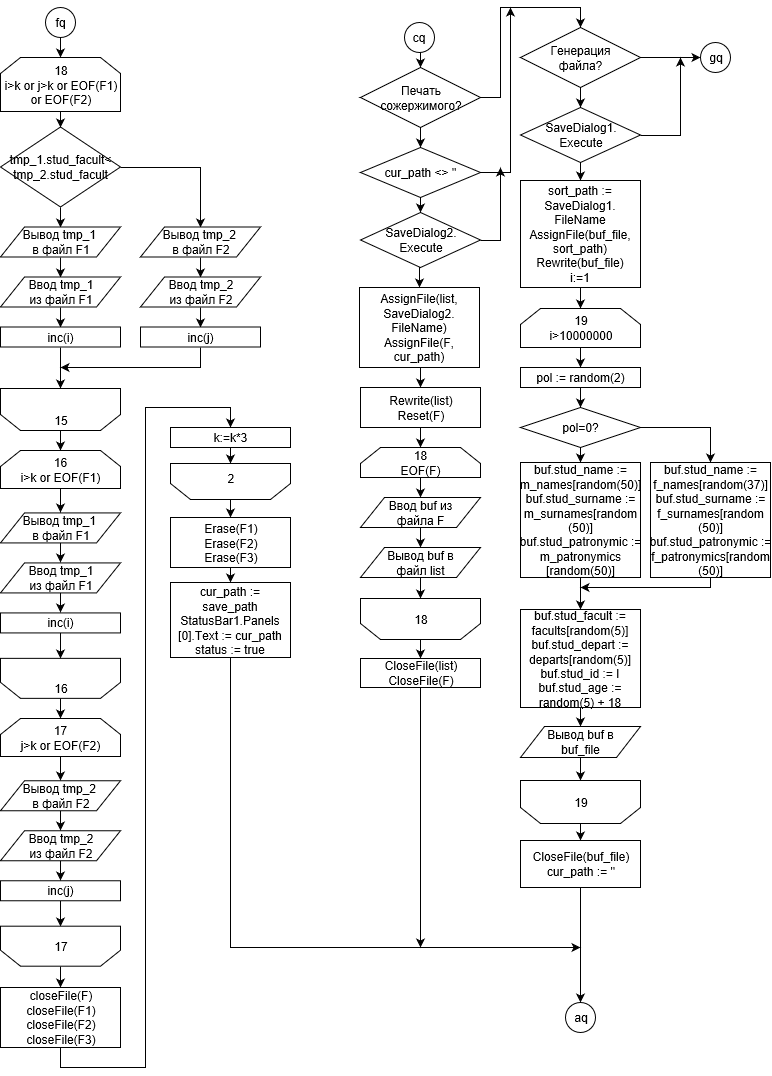
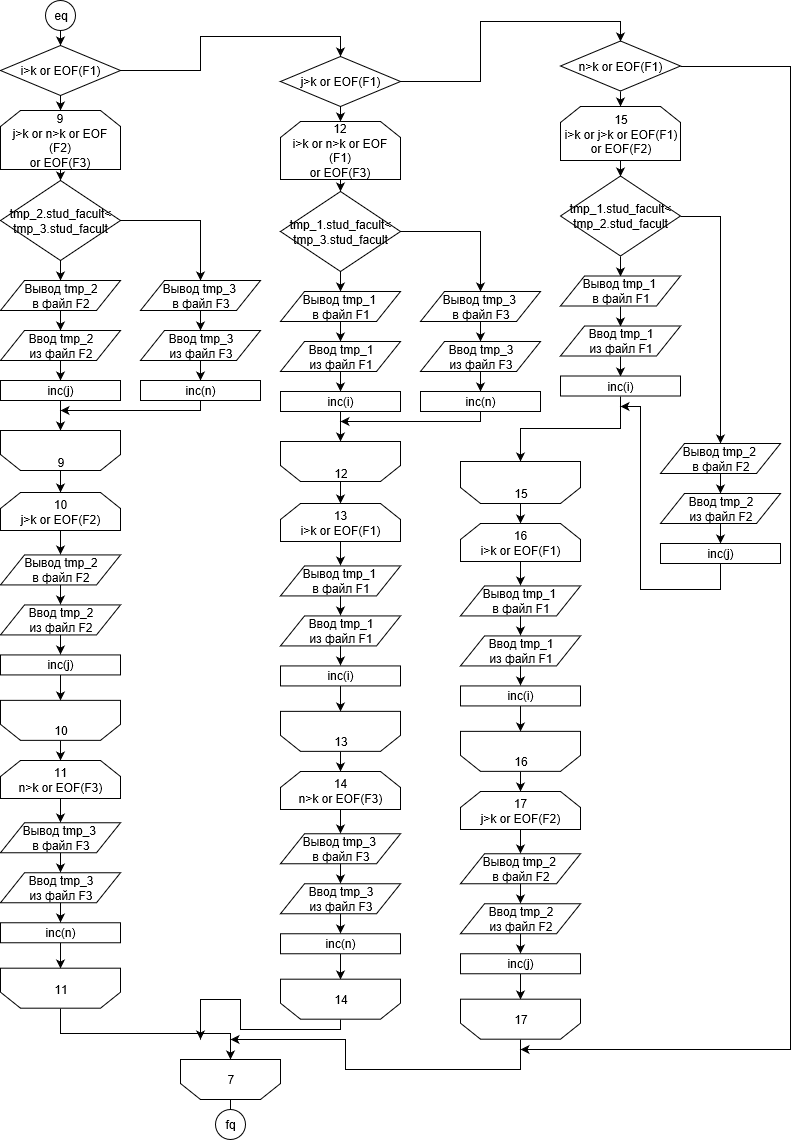
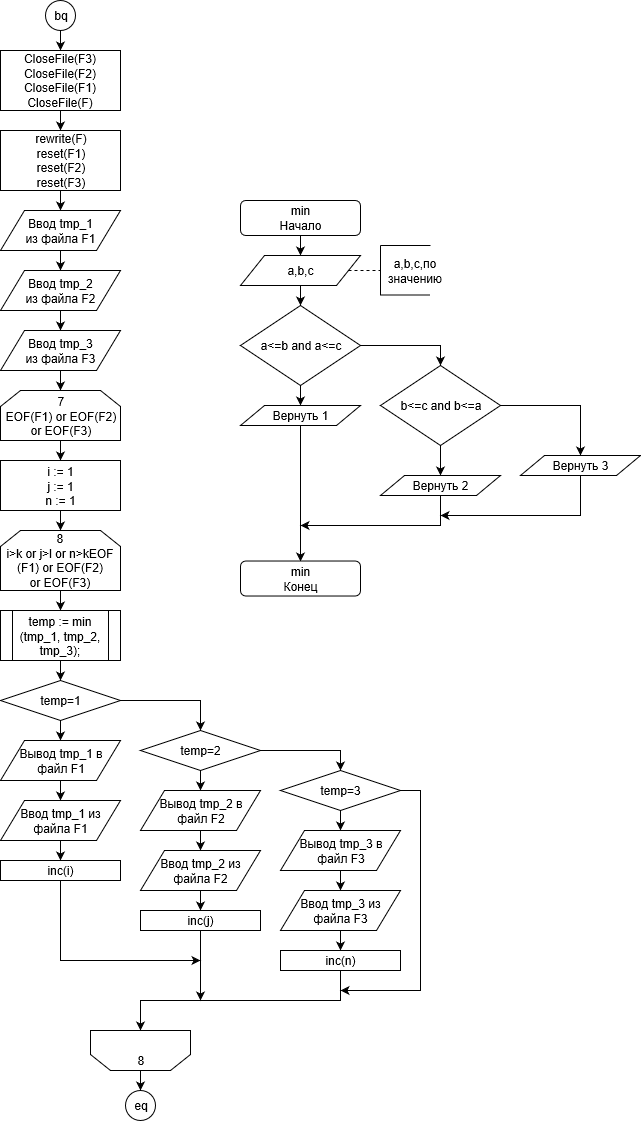
**2.Размер сортируемых данных должен быть не менее 1 Гб. Объем потребляемой оперативной памяти не должен превышать 10% от размера исходного файла.**

**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 алгоритмов:**

****

****

**Исходный текст программы:**

**unit Unit1;**

**{$mode objfpc}{$H+}**

**interface**

**uses**

**Classes, SysUtils, FileUtil, Forms, Controls, Graphics, Dialogs, Menus,**

**ComCtrls, StdCtrls;**

**type**

**{ TForm1 }**

**TForm1 = class(TForm)**

**Button1: TButton;**

**MainMenu1: TMainMenu;**

**MenuItem1: TMenuItem;**

**MenuItem2: TMenuItem;**

**MenuItem3: TMenuItem;**

**MenuItem4: TMenuItem;**

**MenuItem5: TMenuItem;**

**MenuItem6: TMenuItem;**

**MenuItem7: TMenuItem;**

**MenuItem9: TMenuItem;**

**OpenDialog1: TOpenDialog;**

**SaveDialog1: TSaveDialog;**

**SaveDialog2: TSaveDialog;**

**StatusBar1: TStatusBar;**

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

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

**procedure MenuItem2Click(Sender: TObject);**

**procedure MenuItem3Click(Sender: TObject);**

**procedure MenuItem4Click(Sender: TObject);**

**procedure MenuItem7Click(Sender: TObject);**

**procedure MenuItem9Click(Sender: TObject);**

**private**

**public**

**end;**

**type students = record**

**stud\_surname, stud\_name, stud\_patronymic, stud\_facult, stud\_depart : string [40];**

**stud\_id : integer;**

**stud\_age : byte;**

**end;**

**var**

**Form1: TForm1;**

**out\_file, buf\_file, sort\_file : file of students;**

**sort\_path, save\_path, cur\_path : string;**

**m\_names : array [0..49] of string = ('Александр', 'Марк','Георгий','Артемий','Дмитрий','Константин','Давид','Эмиль','Максим','Тимур','Платон','Назар', 'Сергей','Олег','Анатолий','Савва','Андрей','Ярослав','Григорий,'Ян','Алексей','Антон','Демид','Рустам','Артём','Николай','Данила','Игнат','Илья','Глеб','Станислав','Влад','Кирилл','Данил','Василий','Альберт','Михаил','Савелий','Федор','Тамерлан','Никита','Вадим','Родион','Айдар','Матвей','Степан','Леонид','Роберт','Роман','Дамир');**

**f\_names : array [0..36] of string = ('Мария','Марина','Алина',**

**'Олеся','Ирина','Александра','София','Клавдия','Жанна','Елена',**

**'Валерия','Анна','Венера','Анастасия','Екатерина','Любовь',**

**'Светлана','Ксения','Юлия','Оксана','Алёна','Полина','Дана',**

**'Елизавета','Виктория','Людмила','Ева','Лана','Эвелина',**

**'Милена','Милана','Альбина','Нина','Зоя','Кристина','Татьяна',**

**'Наталия');**

**f\_surnames : array [0..49] of string = ('Иванова' ,'Смирнова' ,'Кузнецова','Попова' ,'Васильева' ,'Петрова' ,'Соколова' ,'Михайлова' ,'Новикова' ,'Федорова' ,'Морозова' ,'Волкова','Алексеева' ,'Лебедева' ,'Семенова' ,'Егорова' ,'Павлова','Козлова','Степанова' ,'Николаева' ,'Орлова' ,'Андреева','Макарова' ,'Никитина' ,'Захарова' ,'Зайцева' ,'Соловьева' ,'Борисова' ,'Яковлева' ,'Григорьева' ,'Романова' ,'Воробьева' ,'Сергеева' ,'Кузьмина' ,'Фролова' ,'Александрова','Дмитриева','Королева','Гусева' ,'Киселева' ,'Ильина','Давыдова','Беляева','Герасимова','Богданова',**

**'Осипова','Сидорова','Матвеева','Титова','Маркова');**

**m\_surnames : array [0..49] of string = ('Иванов','Смирнов','Кузнецов','Попов' ,'Васильев' ,'Петров' ,'Соколов','Михайлов' ,'Новиков' ,'Федоров','Морозов','Волков','Алексеев' ,'Лебедев' ,'Семенов' ,'Егоров' ,'Павлов' ,'Козлов' ,'Степанов' ,'Николаев' ,'Орлов' ,'Андреев' ,'Макаров' ,'Никитин' ,'Захаров' ,'Зайцев' ,'Соловьев' ,'Борисов' ,'Яковлев' ,'Григорьев' ,'Романов' ,'Воробьев','Сергеев' ,'Кузьмин' ,'Фролов' ,'Александров' ,'Дмитриев' ,'Королев' ,'Гусев','Киселев' ,'Ильин' ,'Максимов' ,'Поляков' ,'Сорокин' ,'Виноградов' ,'Ковалев' ,'Белов' ,'Медведев' ,'Антонов' ,'Тарасов');**

**m\_patronymics : array [0..49] of string = ('Александрович','Маркович', 'Георгиевич','Артемиевич','Дмитриевич','Константинович','Давидович', 'Эмильевич','Максимович','Тимурович','Платонович','Назарович','Сергеевич', 'Олегович','Анатолиевич','Андреевич','Ярославович','Григориевич','Янович', 'Алексеевич','Антонович','Демидович','Рустамович','Артёмович','Николаевич', 'Данилович','Игнатьевич','Ильич','Глебович','Станиславович','Владиславович', 'Кириллович','Святославович','Ильдарович','Иванович','Семенович','Борисович', 'Самирович','Денисович','Макарович','Эдуардович','Тихонович','Евгениевич', 'Львович','Маратович','Рамильевич','Даниилович','Викторович','Германович','Ренатович');**

**f\_patronymics : array [0..49] of string = ('Александровна','Марковна', 'Георгиевна','Артемиевна','Дмитриевна','Константиновна','Давидовна', 'Эмильевна','Максимовна','Тимуровна','Платоновна','Назаровна','Сергеевна', 'Олеговна','Анатольевна','Андреевна','Ярославовна','Григориевна','Яновна',**

**'Алексеевна','Антоновна','Демидовна','Рустамовна','Артёмовна','Николаевна', 'Даниловна','Игнатьевна','Глебовна','Станиславовна','Владиславовна', 'Кирилловна','Даниловна','Васильевна','Альбертовна','Михаиловна','Савельевна', 'Федоровна','Тамерлановна','Никитична','Вадимовна','Родионовна','Матвеевна',**

**'Степановна','Леонидовна','Робертовна','Романовна','Юриьевна','Егоровна','Богдановна','Валерьевна');**

**facults : array [0..5] of string = ('ФАВТ','ФПМТ','БФ','ХФ','ФКиФМН','ПФ');**

**departs : array [0..5] of string = ('КБ','КФ','КЭВМ','КАиТ','КАиГ','КТиГ');**

**buf : students;**

**status : boolean;**

**implementation**

**{$R \*.lfm}**

**{ TForm1 }**

**procedure TForm1.MenuItem2Click(Sender: TObject);**

**begin**

**if (OpenDialog1.Execute) then begin**

**StatusBar1.Panels[0].Text := OpenDialog1.FileName;**

**cur\_path := OpenDialog1.FileName;**

**end;**

**end;**

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

**begin**

**sort\_path := '';**

**save\_path := '';**

**status := false;**

**end;**

**function min(a, b, c : students) : byte;**

**begin**

**if (a.stud\_facult <= b.stud\_facult) and (a.stud\_facult <= c.stud\_facult) then min := 1 else**

**if (b.stud\_facult <= a.stud\_facult) and (b.stud\_facult <= c.stud\_facult) then min := 2 else min := 3;**

**end;**

**function min2(tmp\_1, tmp\_2 : students) : byte;**

**begin**

**if (tmp\_1.stud\_facult < tmp\_2.stud\_facult) then min2 := 1 else**

**if (tmp\_1.stud\_facult = tmp\_2.stud\_facult) then begin**

**if (tmp\_1.stud\_surname < tmp\_2.stud\_surname) then min2 := 1 else**

**if (tmp\_1.stud\_surname = tmp\_2.stud\_surname) then begin**

**if (tmp\_1.stud\_name < tmp\_2.stud\_name) then min2 := 1 else**

**if (tmp\_1.stud\_name = tmp\_2.stud\_name) then begin**

**if (tmp\_1.stud\_patronymic < tmp\_2.stud\_patronymic) then min2 := 1 else**

**if (tmp\_1.stud\_patronymic = tmp\_2.stud\_patronymic) then min2 := 1 else min2 := 2;**

**end else min2 := 2;**

**end else min2 := 2;**

**end else min2 := 2;**

**end;**

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

**Var count, k, i, j, n: integer;**

**temp : byte;**

**F, F1, F2, F3, F4: file of students;**

**tmp\_1, tmp\_2, tmp\_3 : students;**

**begin**

**if (cur\_path <> '') then begin**

**if (SaveDialog1.Execute) then begin**

**save\_path := SaveDialog1.FileName;**

**if (save\_path <> cur\_path) then begin**

**AssignFile(F, save\_path);**

**AssignFile(F4, cur\_path);**

**AssignFile(F1, 'file1.swm');**

**AssignFile(F2, 'file2.swm');**

**AssignFile(F3, 'file3.swm');**

**count := 0;**

**reset(F4);**

**rewrite(F);**

**while (not EOF(F4)) do begin**

**read(F4, tmp\_1);**

**write(F, tmp\_1);**

**inc(count);**

**end;**

**CloseFile(F);**

**CloseFile(F4);**

**k := 1;**

**i:= 1; j := 1; n := 1;**

**while (k < count) do begin**

**i:= 1; j := 1; n := 1;**

**reset(F);**

**rewrite(F1);**

**rewrite(F2);**

**rewrite(F3);**

**while (not EOF(F)) do begin**

**while (i <= k) and (not EOF(F)) do begin**

**read(F, tmp\_1);**

**write(F1, tmp\_1);**

**inc(i);**

**end;**

**while (j <= k) and (not EOF(F)) do begin**

**read(F, tmp\_1);**

**write(F2, tmp\_1);**

**inc(j);**

**end;**

**while (n <= k) and (not EOF(F)) do begin**

**read(F, tmp\_1);**

**write(F3, tmp\_1);**

**inc(n);**

**end;**

**i := 1;**

**j := 1;**

**n := 1;**

**end;**

**tmp\_1.stud\_id := -1;**

**write(F1, tmp\_1);**

**write(F2, tmp\_1);**

**write(F3, tmp\_1);**

**CloseFile(F3);**

**CloseFile(F2);**

**CloseFile(F1);**

**CloseFile(F);**

**rewrite(F);**

**reset(F1);**

**reset(F2);**

**reset(F3);**

**if (not EOF(F1)) then read(F1, tmp\_1);**

**if (not EOF(F2)) then read(F2, tmp\_2);**

**if (not EOF(F3)) then read(F3, tmp\_3);**

**while (not EOF(F1)) and (not EOF(F2)) and (not EOF(F3)) do begin**

**i := 1; j := 1; n := 1;**

**if (tmp\_2.stud\_id = -1) then tmp\_2.stud\_id := 1000000;**

**if (tmp\_3.stud\_id = -1) then tmp\_3.stud\_id := 1000000;**

**while (i <= k) and (j <= k) and (n <= k) and (not EOF(F1)) and (not EOF(F2)) and (not EOF(F3)) do begin**

**temp := min2(tmp\_1, tmp\_2);**

**if (temp = 1) then begin**

**temp := min2(tmp\_1, tmp\_3);**

**if (temp = 2) then temp := 3 else temp := 1;**

**end else begin**

**temp := min2(tmp\_2, tmp\_3);**

**if (temp = 2) then temp := 3 else temp := 2;**

**end;**

**case temp of**

**1 : begin**

**write(F, tmp\_1);**

**read(F1, tmp\_1);**

**inc(i);**

**end;**

**2 : begin**

**write(F, tmp\_2);**

**read(F2, tmp\_2);**

**inc(j);**

**end;**

**3 : begin**

**write(F, tmp\_3);**

**read(F3, tmp\_3);**

**inc(n);**

**end;**

**end;**

**end;**

**if (i > k) or (i > k ) and (EOF(F1)) then begin**

**while (j <= k) and (n <= k) and (not EOF(F2)) and (not EOF(F3)) do begin**

**temp := min2(tmp\_2, tmp\_3);**

**case temp of**

**1 : begin**

**write(F, tmp\_2);**

**read(F2, tmp\_2);**

**inc(j);**

**end;**

**2 : begin**

**write(F, tmp\_3);**

**read(F3, tmp\_3);**

**inc(n);**

**end;**

**end;**

**end;**

**while (j <= k) and (not EOF(F2)) do begin**

**write(F, tmp\_2);**

**read(F2, tmp\_2);**

**inc(j);**

**end;**

**while (n <= k) and (not EOF(F3)) do begin**

**write(F, tmp\_3);**

**read(F3, tmp\_3);**

**inc(n);**

**end**

**end else if (j > k) or (j > k) and (EOF(F2)) then begin**

**while (i <= k) and (n <= k) and (not EOF(F1)) and (not EOF(F3)) do begin**

**temp := min2(tmp\_1, tmp\_3);**

**case temp of**

**1 : begin**

**write(F, tmp\_1);**

**read(F1, tmp\_1);**

**inc(i);**

**end;**

**2 : begin**

**write(F, tmp\_3);**

**read(F3, tmp\_3);**

**inc(n);**

**end;**

**end;**

**end;**

**while (i <= k) and (not EOF(F1)) do begin**

**write(F, tmp\_1);**

**read(F1, tmp\_1);**

**inc(i);**

**end;**

**while (n <= k) and (not EOF(F3)) do begin**

**write(F, tmp\_3);**

**read(F3, tmp\_3);**

**inc(n);**

**end;**

**end else if (n > k) or (n > k) and (EOF(F3)) then begin**

**while (i <= k) and (j <= k) and (not EOF(F1)) and (not EOF(F2)) do begin**

**temp := min2(tmp\_1, tmp\_2);**

**case temp of**

**1 : begin**

**write(F, tmp\_1);**

**read(F1, tmp\_1);**

**inc(i);**

**end;**

**2 : begin**

**write(F, tmp\_2);**

**read(F2, tmp\_2);**

**inc(j);**

**end;**

**end;**

**end;**

**while (i <= k) and (not EOF(F1)) do begin**

**write(F, tmp\_1);**

**read(F1, tmp\_1);**

**inc(i);**

**end;**

**while (j <= k) and (not EOF(F2)) do begin**

**write(F, tmp\_2);**

**read(F2, tmp\_2);**

**inc(j);**

**end;**

**end;**

**end;**

**if (tmp\_3.stud\_id = 1000000) or (tmp\_3.stud\_id = -1) and (tmp\_2.stud\_id <> -1) then begin**

**i := 1; j := 1;**

**while (i <= k) and (j <= k) and (not EOF(F1)) and (not EOF(F2)) do begin**

**temp := min2(tmp\_1, tmp\_2);**

**case temp of**

**1 : begin**

**write(F, tmp\_1);**

**read(F1, tmp\_1);**

**inc(i);**

**end;**

**2 : begin**

**write(F, tmp\_2);**

**read(F2, tmp\_2);**

**inc(j);**

**end;**

**end;**

**end;**

**while (i <= k) and (not EOF(F1)) do begin**

**write(F, tmp\_1);**

**read(F1, tmp\_1);**

**inc(i);**

**end;**

**while (j <= k) and (not EOF(F2)) do begin**

**write(F, tmp\_2);**

**read(F2, tmp\_2);**

**inc(j);**

**end;**

**end else if (tmp\_2.stud\_id = -1) and (tmp\_3.stud\_id = -1) then**

**while (not EOF(F1)) do begin**

**write(F, tmp\_1);**

**read(F1, tmp\_1);**

**end;**

**closeFile(F);**

**closeFile(F1);**

**closeFile(F2);**

**closeFile(F3);**

**k := k \* 3;**

**end;**

**Erase(F1);**

**Erase(F2);**

**Erase(F3);**

**ShowMessage('Файл отсортирован');**

**cur\_path := save\_path;**

**StatusBar1.Panels[0].Text := cur\_path;**

**status := true;**

**end else ShowMessage('Пути файлов не должны совпадать');**

**end;**

**end else ShowMessage('Файл не выбран');**

**end;**

**procedure TForm1.MenuItem3Click(Sender: TObject);**

**var list : text;**

**F : file of students;**

**begin**

**if (cur\_path <> '') then begin**

**if (SaveDialog2.Execute) then begin**

**AssignFile(list, SaveDialog2.FileName);**

**AssignFile(F, cur\_path);**

**Rewrite(list);**

**Reset(F);**

**while (not EOF(F)) do begin**

**read(F, buf);**

**writeln(list, buf.stud\_surname, ' ', buf.stud\_name,' ', ' ', buf.stud\_patronymic,**

**' ', buf.stud\_facult, ' ', buf.stud\_depart, ' ', buf.stud\_id, ' ', buf.stud\_age);**

**end;**

**ShowMessage('Сожержимое файла напечатано');**

**CloseFile(list);**

**CloseFile(F);**

**end;**

**end else ShowMessage('Файл не выбран');**

**end;**

**procedure TForm1.MenuItem4Click(Sender: TObject);**

**begin**

**close();**

**end;**

**procedure TForm1.MenuItem7Click(Sender: TObject);**

**var i : longint;**

**pol : byte;**

**begin**

**if (SaveDialog1.Execute) then begin**

**randomize;**

**sort\_path := SaveDialog1.FileName;**

**AssignFile(buf\_file, sort\_path);**

**Rewrite(buf\_file);**

**for i := 1 to {4782969} 40 do begin**

**pol := random(2);**

**if (pol = 0) then begin**

**buf.stud\_name := m\_names[random(50)];**

**buf.stud\_surname := m\_surnames[random(50)];**

**buf.stud\_patronymic := m\_patronymics[random(50)];**

**end else begin**

**buf.stud\_name := f\_names[random(37)];**

**buf.stud\_surname := f\_surnames[random(50)];**

**buf.stud\_patronymic := f\_patronymics[random(50)];**

**end;**

**buf.stud\_facult := facults[random(5)];**

**buf.stud\_depart := departs[random(5)];**

**buf.stud\_id := i;**

**buf.stud\_age := random(5) + 18;**

**write(buf\_file, buf);**

**end;**

**CloseFile(buf\_file);**

**ShowMessage('Файл успешно создан');**

**cur\_path := '';**

**end;**

**end;**

**procedure TForm1.MenuItem9Click(Sender: TObject);**

**var F : file of students;**

**buf\_1, buf\_2 : students;**

**flag : boolean;**

**temp : byte;**

**begin**

**if (cur\_path <> '') then begin**

**flag := false;**

**AssignFile(F, cur\_path);**

**Reset(F);**

**if (not EOF(F)) then read(F, buf\_1);**

**while (not EOF(F)) do begin**

**read(F, buf\_2);**

**temp := min2(buf\_1, buf\_2);**

**if (temp = 1) then begin**

**buf\_1 := buf\_2;**

**end else begin**

**ShowMessage('Файл не является отсортированным');**

**flag := true;**

**break;**

**end;**

**end;**

**if (not flag) then ShowMessage('Файл является отсортированным');**

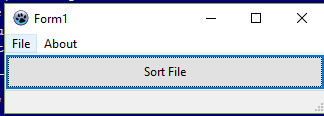
**CloseFile(F);**

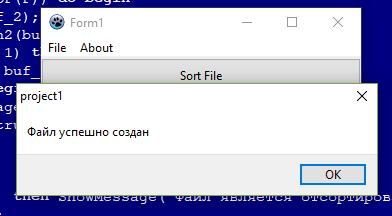
**end else ShowMessage('Файл не выбран');**

**end;**

**end.**

**Скриншоты выполнения программы**

****

****

**Вывод**

В ходе данной лабораторной работы были получены знания для работы со средой Lazarus. Освоена работа с организацией стандартных диалоговых окон, изучены принципы работы с данными на внешней памяти, получены навыки работы с типизированными файлами. Была создана программа для сортировки данных, находящихся на жестком диске в типизированном файле. Были изучены принципы событийно-ориентированного программирования, получены навыки создания программ с графическим интерфейсом пользователя. Размер сортируемых данных не менее 1 Гб. Объем потребляемой оперативной памяти не превышает 10% от размера исходного файла. Исходный и целевой файлы выбираются с помощью стандартных диалоговых окон. Также создана структура сортируемого файла. Помимо основной программы, реализованы две вспомогательные утилиты – для генерации массива данных и проверки упорядоченности.

Получен практический навык составления схем-алгоритмов с применением следующих символов:

1. цикл;

2. процесс;

3. данные;

4. решение;

5. терминальный символ.

Лабораторная работа выполнялась в бесплатной среде разработки Lazarus. Данная среда предоставляет много возможностей, в числе которых есть отладчик, который применялся в процессе написания программы для её отладки. Таким образом, был получен навык работы с отладчиком и поиск ошибок в программе.