Отчёт по лабораторной работе №7

Тема: Работа со сложными типами данных.

Задание (вариант 2): С помощью программных средств С/C++ создать мини-СУБД, используя заданную вариантом структуру данных.

Программа должна уметь выполнять следующие действия:

* работать в интерактивном режиме с простым и дружественным интерфейсом;
* вводить новые и корректировать имеющиеся данные в СУБД;
* сохранять или не сохранять изменения данных в дисковом файле;
* производить чтение сохраненных ранее данных СУБД из дискового файла;

Вариант структуры базы данных:

ФИО акционера;

Паспорт: серия,номер;

Число имеющихся акций;

Для каждой акции:

Серия,

Номер,

Номинальная стоимость,

Дата покупки: год, месяц, день;

Выполнил:

Студент

Вискунов Дмитрий Юрьевич

КОБ-22-1с

**Постановка задач**

1.Создание структуры базы данных и алгоритма работы с базой данных.

2.Создание функций для работы с базой данных.

3. Создание функции вывода базы данных.

**Структура программы**

1.Подключение необходимых библиотек в заголовочном файле functions.h. В данной программе используются следующие библиотеки: stdio.h – библиотека для работы с потоками данных; iostream – стандартная библиотека для работы с вводом-выводом; fstream – библиотека для работы с записью данных в файлы; string – стандартная библиотека для работы со строками.

2. В файла functions.h задаются прототипы для всех функций программы, сами же функции написаны в файле functions.cpp.

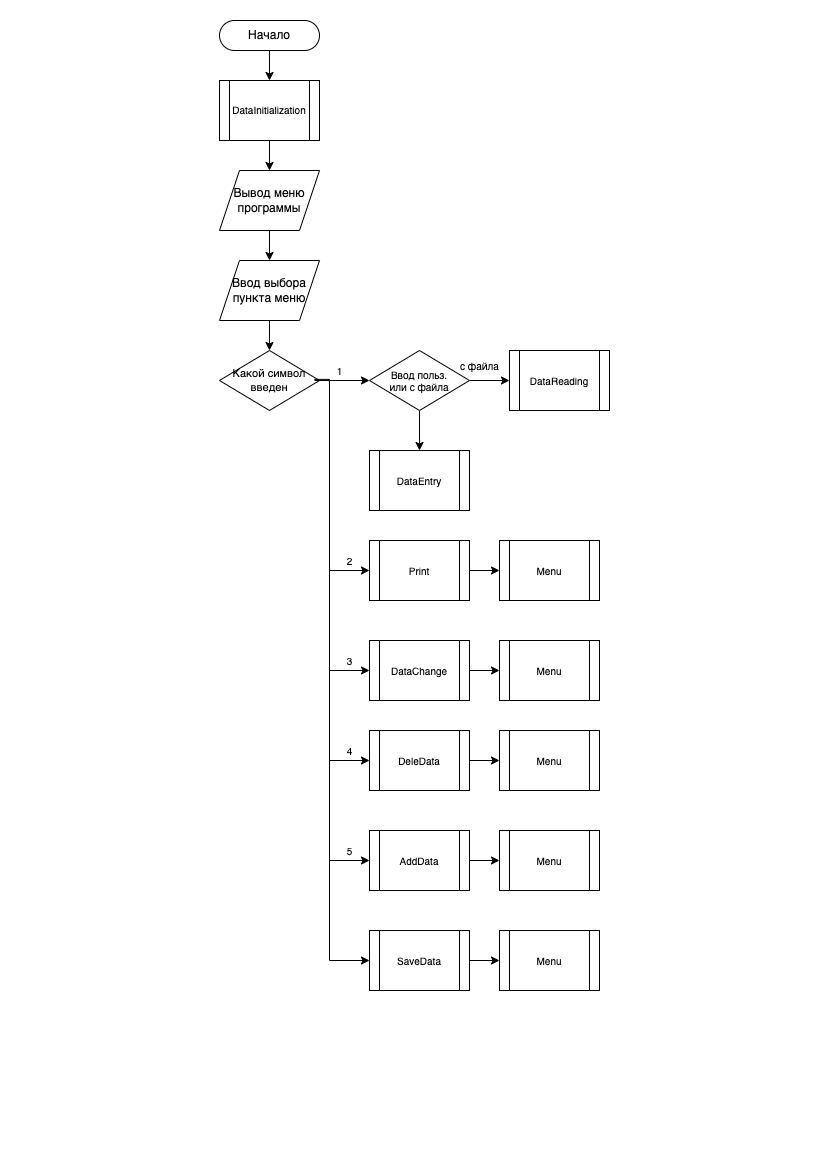
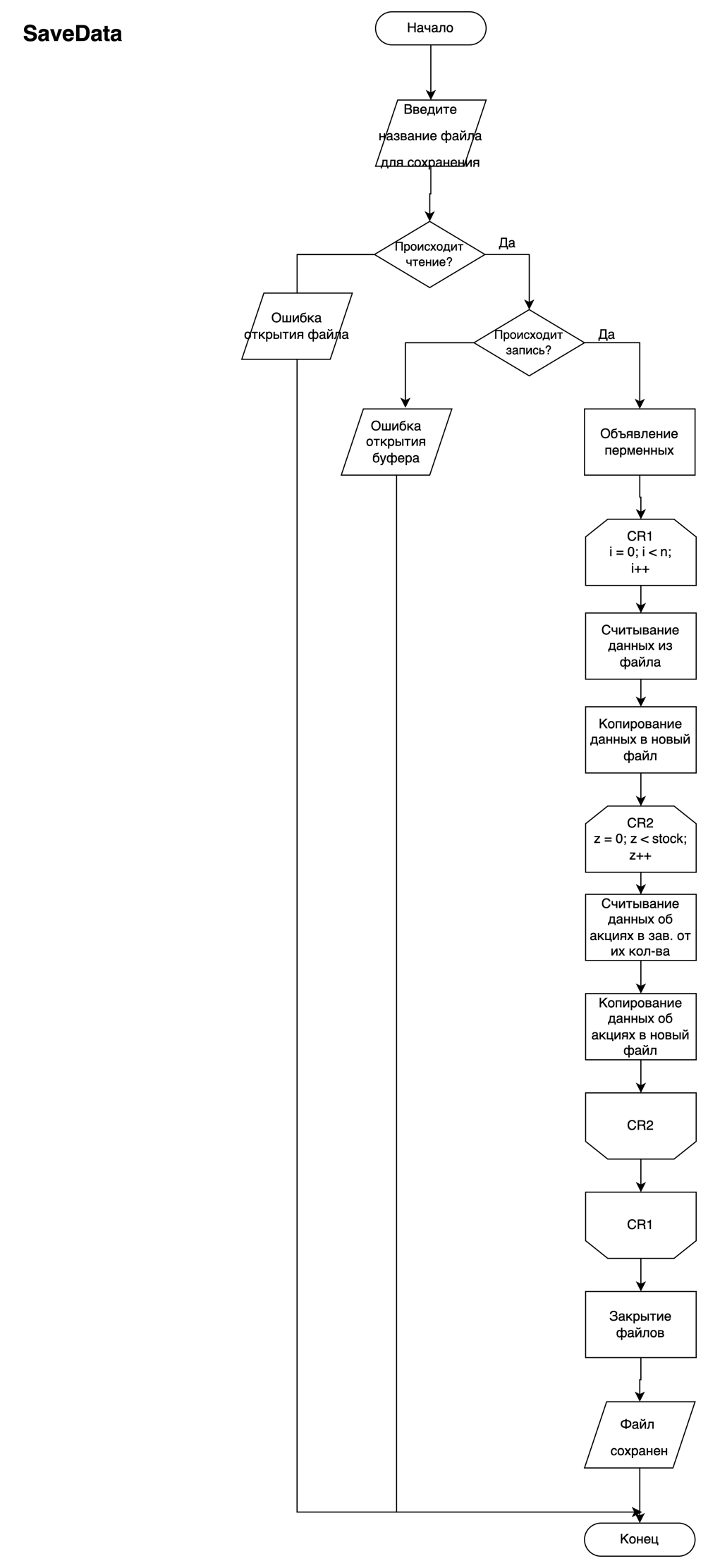
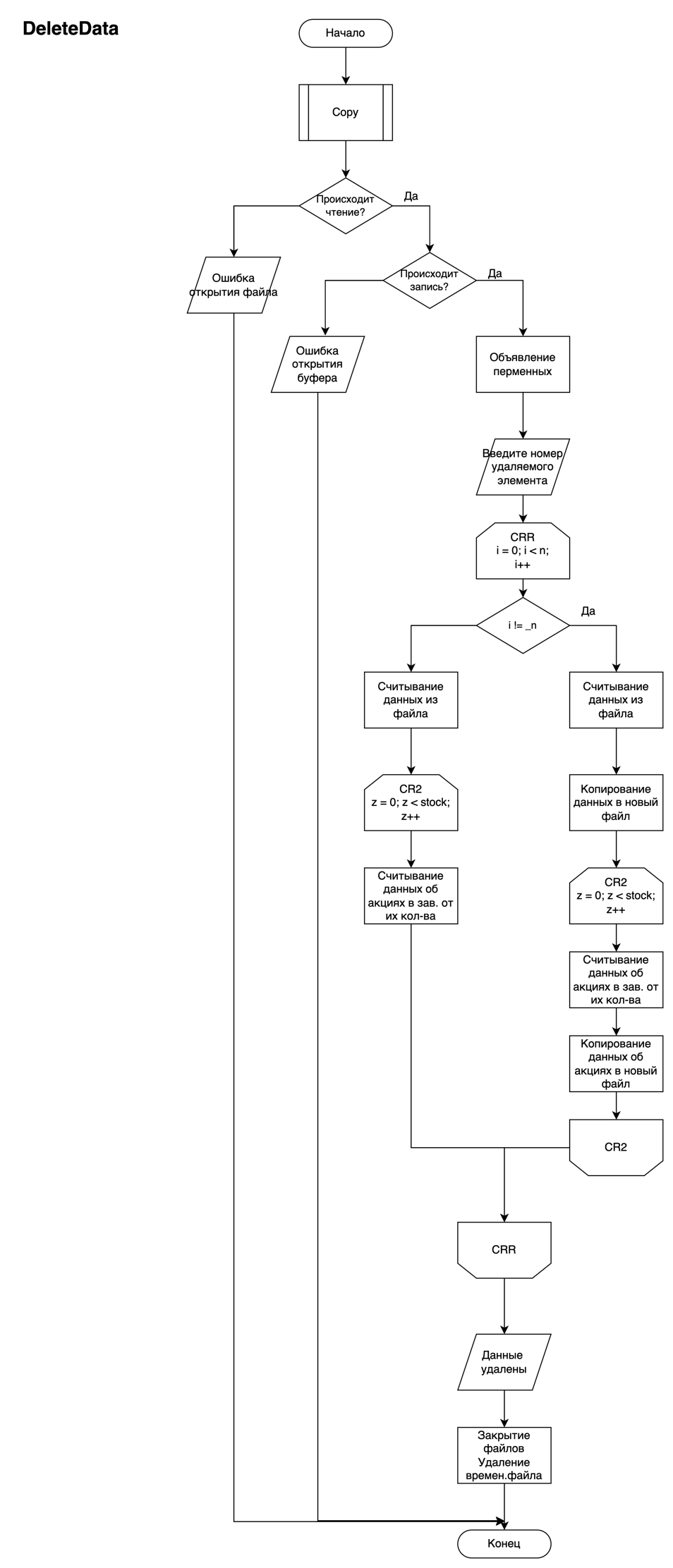
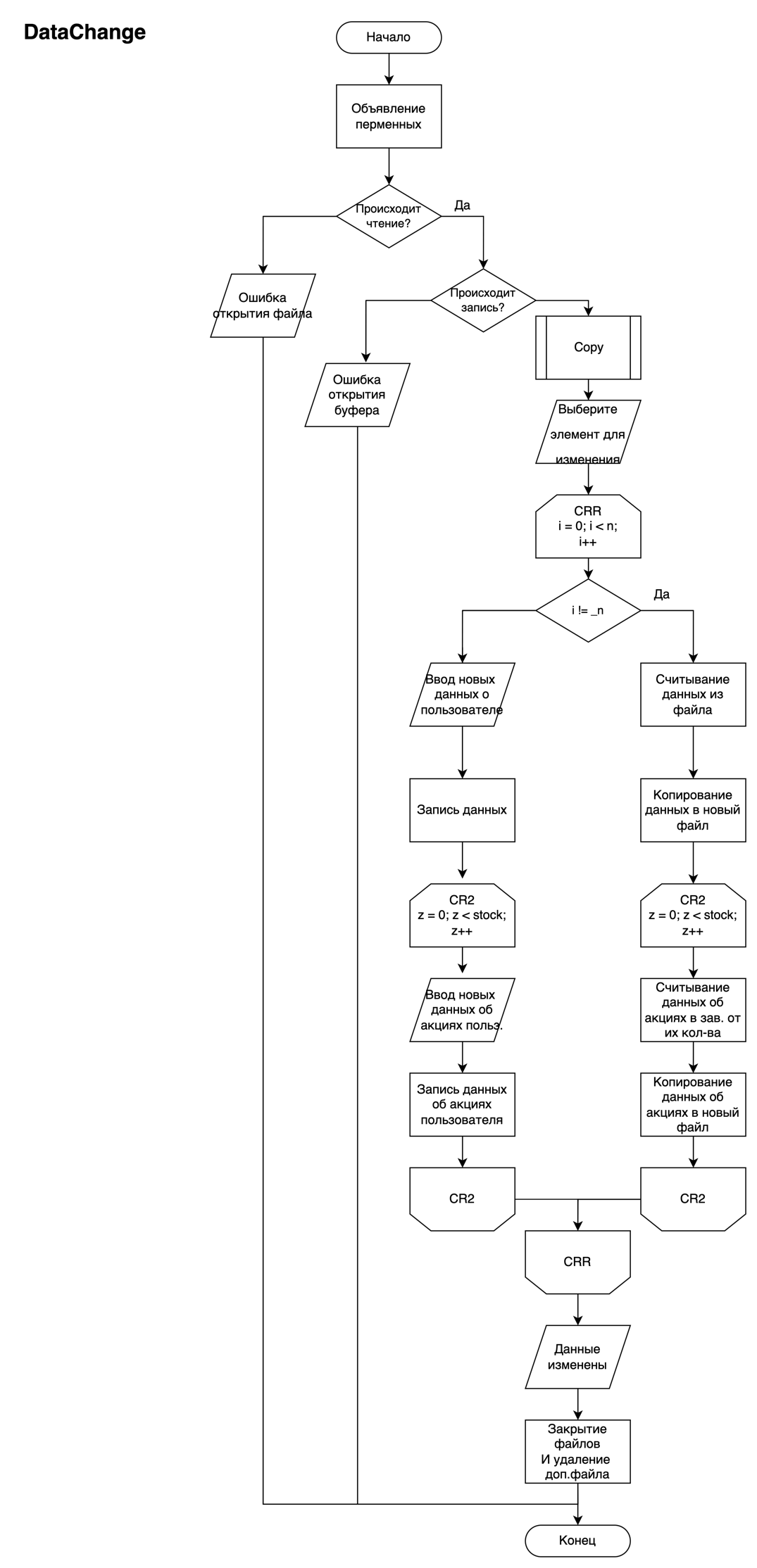
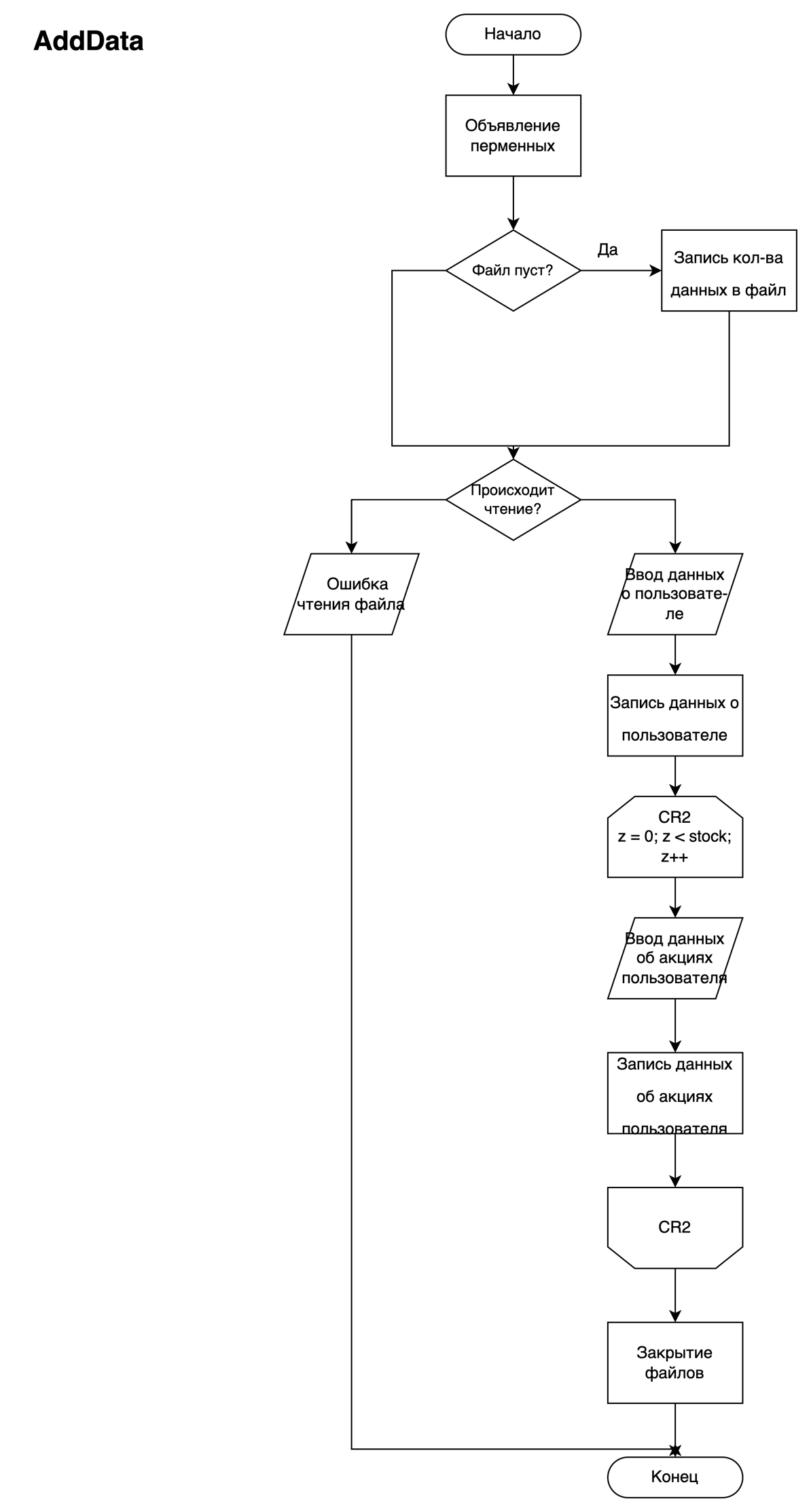
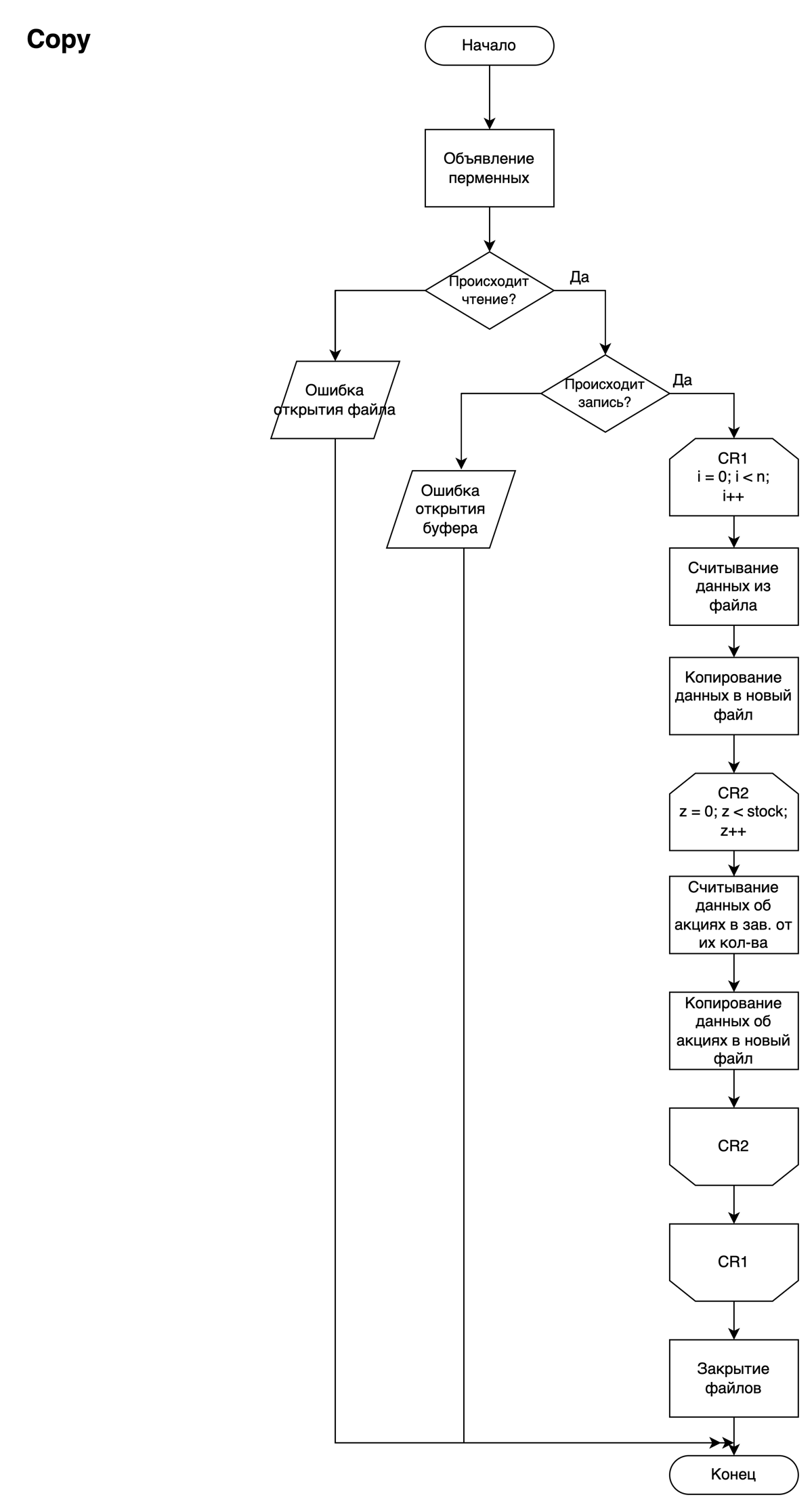
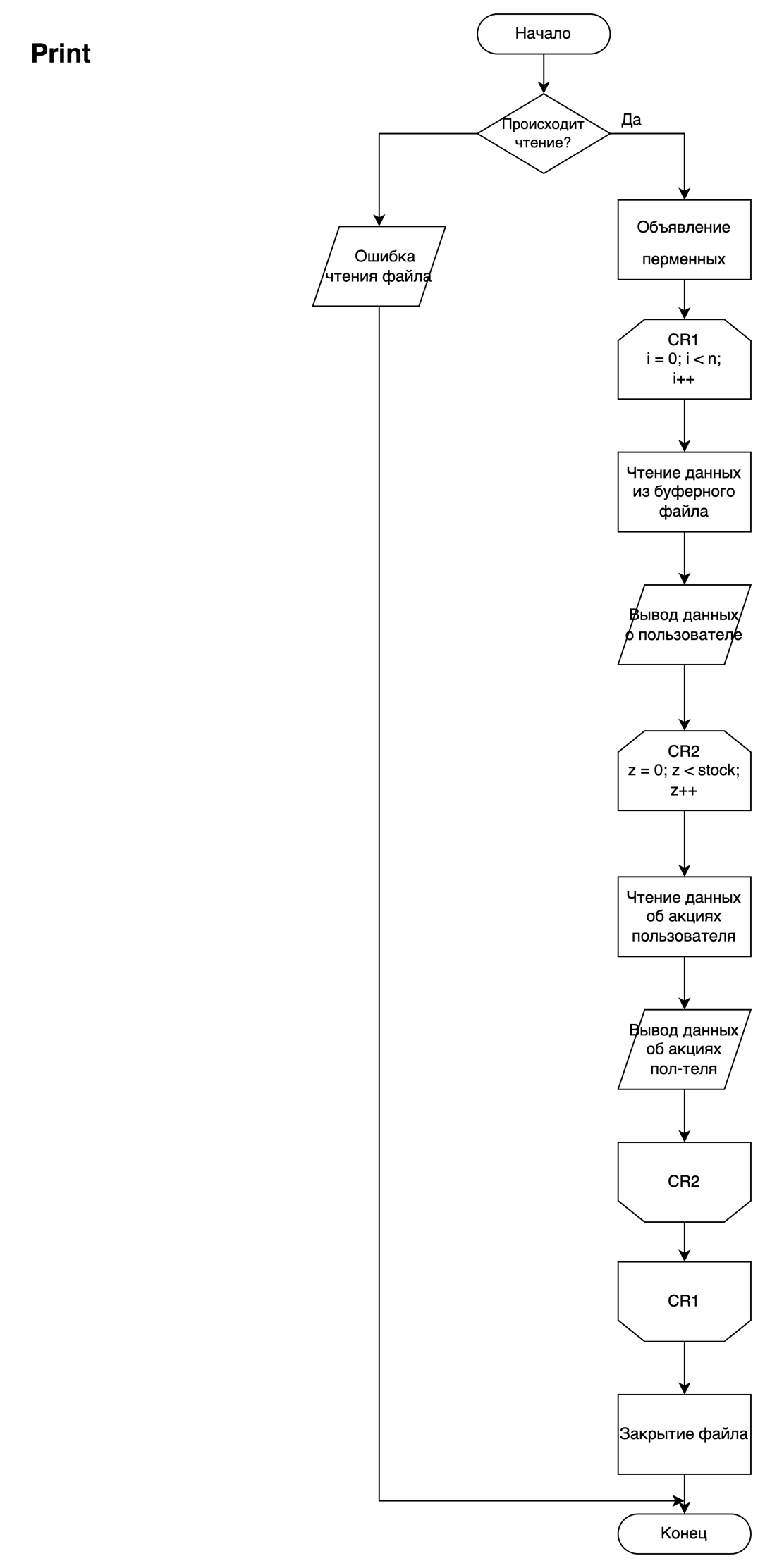
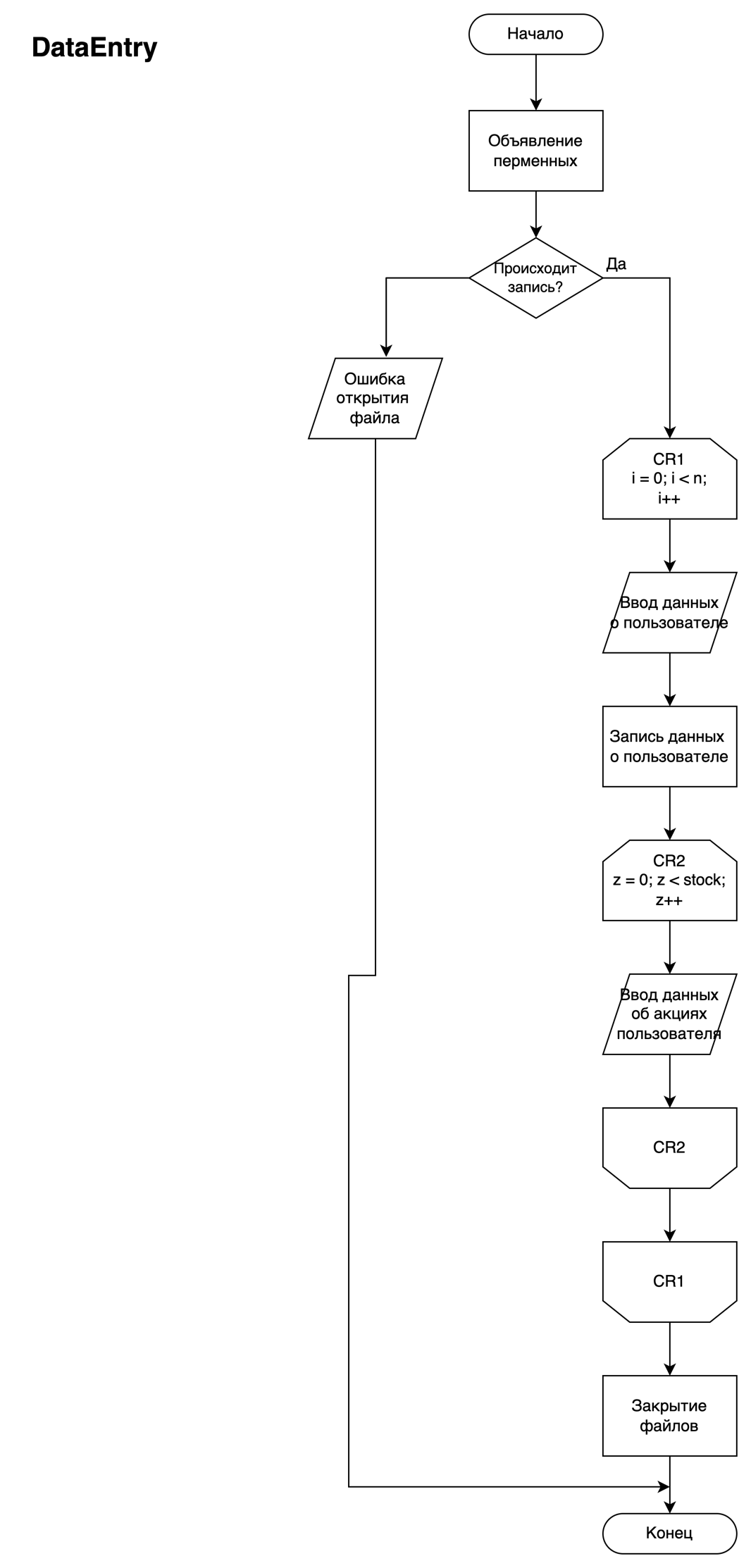
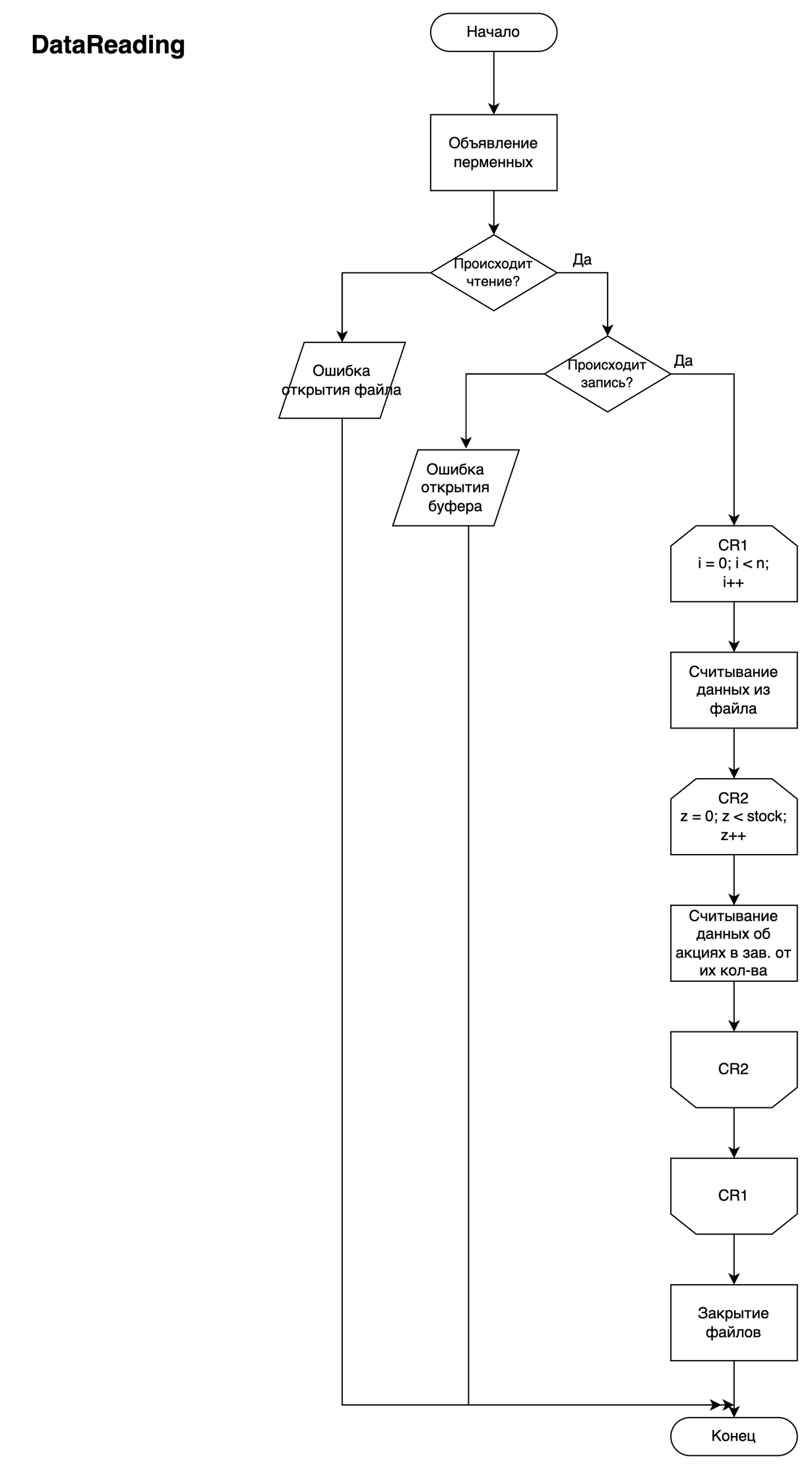
3.В файле main.cpp выводится главное меню программы и создается буферный файл для данных. Меню программы реализовано с помощью конструкции switch-case. Так же в этом файле написан алгоритм проверки очистки данных перед завершением работы программы.

4.В файле functions.cpp описаны все функции программы. Функция DataInitialization используется для проверки существования буферного файла необходимого для работы программы. Функция DataEntry используется для ручного ввода данных в программу, в . Функция DataReading используется для чтения данных из файла и записи в буферный файл программы. Функция Print используется для вывода БД на экран. Функция DataCleaning очищает буферный файл программы и возвращает резултат работы где соотвественно 0 – файл не был очищен, а 1 – файл очищен. Функция DataChange используется для изменения конкретных данных БД. Функция AmountOfData используется для обозначения общего количества данных в файле при использовании функции AddData. Функция DeleteData удаляет конкретный элемент БД. Функция AddData добавляет новый элемент в БД и соответственно структуре БД (первая строка в БД указывает на количество данных) увеличивает количество данных. Функция SaveData используется для сохранения БД в отдельный файл.

5. Все функции данной программы реализуются с помощью записи и чтения из файла буфера.

6.Для каждой функции написан вывод ошибок для более удобной работы с программой. Так же в каждой функции для корректной работы прописывается отдельно открытие файлов и их закрытие по окончанию работы с функцией.

**Графическое описание алгоритма**

****

**Код программы**

**functions.h**

#pragma once // защита от двойного подключения файла

#include <stdio.h> // библиотека работы с потоками

#include <iostream> // стандартная библиотека ввода-вывода

#include <fstream> // библиотека для работы с файлами(чтение/запись)

#include <string> // библиотека для работы со строками

using namespace std;

// прототипы функций

void DataInitialization();

void DataEntry();

void DataReading(string FileName);

void Print();

bool DataCleaning();

void DataChange();

int AmountOfData();

void DeleteData();

void AddData();

void SaveData(string FileName);

**main.cpp**

// подключение заголовчного файла с прототипами функций

#include "functions.h"

int \_StateMenu;

void Menu(){

cout << "Выберите действие:" << endl;

cout << "(0) Выход из программы" << endl;

cout << "(1) Ввод данных" << endl;

cout << "(2) Вывод данных" << endl;

cout << "(3) Изменение данных" << endl;

cout << "(4) Удаление данных" << endl;

cout << "(5) Добавление данных" << endl;

cout << "(6) Сохранение данных" << endl;

cin >> \_StateMenu;

}

int main() {

DataInitialization();// стандартный процесс создания буферного файла

Menu(); // ввыод меню

int \_actions;

string FileName;

while (\_StateMenu !=0){

switch(\_StateMenu){

case 1:

cout << "Ввод вручную или с файла?: ";

cin >> \_actions;

if (\_actions == 1){

// ввод вручную

DataEntry();

}else{

// чтение из файла

cout << "Введите название файла: ";

cin >> FileName;

DataReading(FileName);

}

Menu();

break;

case 2:

Print();

Menu();

break;

case 3:

DataChange();

Menu();

break;

case 4:

DeleteData();

Menu();

break;

case 5:

AddData();

Menu();

break;

case 6:

SaveData(FileName);

Menu();

break;

default:

cout << "Неправильный выбор меню" << endl;

Menu();

break;

}

}

if (DataCleaning()){

cout << "Данные очищены!" << endl;

}else{

cout << "Данные НЕ очищены!" << endl;

}

cout << "Работа завершена" << endl;

}

**functions.cpp**

#include "functions.h" // подключение заголовчного файла с прототипами функций

using namespace::std;

void DataInitialization(){

// создаем буферный файл

ofstream \_buf("Buffer.txt");

if(!\_buf){

cout << "Ошибка создания буферного файла!" << endl;}

\_buf.close();

}

int AmountOfData(){

ifstream \_buf("Buffer.txt");

int n;

if(\_buf){

\_buf >> n;

}else{cout << "Ошибка чтения файла" << endl;}

\_buf.close();

return n;

}

void DataEntry(){

// временные переменные

string \_surname;

string \_name;

string \_patronymic;

int stock;

int stockSerial[99];

int stockNumber[99];

int stockValue[99];

string DateOfBuy[99];

int pasport;

int n;

cout << "Введите количество данных: ";

cin >> n;

ofstream record("Buffer.txt",ios::app);

// запись данных по количеству

if(record){

record << n << endl;

for(int i = 0; i < n;i++){

cout << "Введите фамилию: ";

cin >> \_surname;

record << \_surname << endl;

cout << "Введите имя: ";

cin >> \_name;

record << \_name << endl;

cout << "Введите отчество: ";

cin >> \_patronymic;

record << \_patronymic << endl;

cout << "Введите количество акций: ";

cin >> stock;

record << stock << endl;

cout << "Введите номер и серию паспорта: ";

cin >> pasport;

record << pasport << endl;

for(int z = 0; z < stock;z++){

cout << "Введите серию " << z+1 <<" акции: "<< endl;

cin >> stockSerial[z];

record << stockSerial[z] << endl;

cout << "Введите номер" << z+1 <<" акции: " << endl;

cin >> stockNumber[z];

record << stockNumber[z] << endl;

cout << "Введите стоимость" << z+1 <<" акции: " << endl;

cin >> stockValue[z];

record << stockValue[z] << endl;

cout << "Введите дату покупки " << z + 1 <<" акции: "<<\ endl;

cin >> DateOfBuy[z];

record << DateOfBuy[z] << endl;

}

cout <<\ "\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_" << endl;

}

}else{

cout << "Ошибка открытия файла" << endl;

}

record.close();

}

void DataReading(string FileName){

ifstream reading(FileName);

ofstream record("Buffer.txt",ios::out);

if(reading){

if(record){

// временные переменные

string \_surname;

string \_name;

string \_patronymic;

int pasport;

int stock;

int stockSerial[99];

int stockNumber[99];

int stockValue[99];

string DateOfBuy[99];

int n;

// считываем данные

reading >> n;

record << n << endl;

for(int i = 0; i < n;i++){

reading >> \_surname;

reading >> \_name;

reading >> \_patronymic;

reading >> stock;

reading >> pasport;

for(int z = 0; z < stock;z++){

reading >> stockSerial[z];

reading >> stockNumber[z];

reading >> stockValue[z];

reading >> DateOfBuy[z];

}

}

cout << "Данные считаны" << endl;

}else{

cout << "Ошибка открытия буфера" << endl;

}

}else{

cout << "Ошибка открытия файла" << endl;

}

// закрываем файл

reading.close();

record.close();

}

void Print(){

ifstream reading("Buffer.txt");

if(reading){

// временные переменные

int x;

string \_surname;

string \_name;

string \_patronymic;

int stock;

int stockSerial[99];

int stockNumber[99];

int stockValue[99];

string DateOfBuy[99];

int pasport;

int n;

cout << "Введите количество данных: ";

cin >> n;

reading >> x;

for(int i = 0; i < n;i++){

cout << "Данные о владельце №" << i + 1 << endl;

reading >> \_surname;

cout << "Фамилия:" << \_surname << endl;

reading >> \_name;

cout << "Имя:" << \_name << endl;

reading >> \_patronymic;

cout << "Отчество:" << \_patronymic << endl;

reading >> stock;

cout << "Количество акций:" << stock << endl;

reading >> pasport;

cout << "Серия и номер паспорта:" << pasport << endl;

for(int z = 0; z < stock; z++){

cout <<\ "//////////////////////////"<<z+1<<"//////////////////////////" << endl;

reading >> stockSerial[z];

cout << "Серия акции №" << z+1 <<":" <<stockSerial[z]\ << endl;

reading >> stockNumber[z];

cout << "Номер акции №" << z+1 <<":" <<stockNumber[z]\ << endl;

reading >> stockValue[z];

cout << "Стоимость акции №" << z+1 <<":"\ <<stockValue[z]<< endl;

reading >> DateOfBuy[z];

cout << "Дата покупки акции №" << z+1 <<":"\ <<DateOfBuy[z]<< endl;

}

cout << "\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_" << endl;

}

}else{

cout << "Ошибка чтения данных" << endl;

}

reading.close();

}

void Copy(){

ifstream reading("Buffer\_.txt");

ofstream record("Buffer.txt",ios::out);

if(reading){

if(record){

// временные переменные

string \_surname;

string \_name;

string \_patronymic;

int \_stock;

int \_stockSerial[99];

int \_stockNumber[99];

int \_stockValue[99];

string \_DateOfBuy[99];

int \_pasport;

int n;

// читаем данные из буфера в новый

reading >> n;

record << n << endl;

for(int i = 0;i < n;i++){

reading >> \_surname;

record << \_surname << endl;

reading >> \_name;

record << \_name << endl;

reading >> \_patronymic;

record << \_patronymic << endl;

reading >> \_stock;

record << \_stock << endl;

reading >> \_pasport;

record << \_pasport << endl;

for(int z = 0; z < \_stock;z++){

reading >> \_stockSerial[z];

record << \_stockSerial[z] << endl;

reading >> \_stockNumber[z];

record << \_stockNumber[z] << endl;

reading >> \_stockValue[z];

record << \_stockValue[z] << endl;

reading >> \_DateOfBuy[z];

record << \_DateOfBuy[z] << endl;

}

}

}else{cout << "Ошибка открытия файла" << endl;}

}else{cout << "Ошибка открытия буферного файла"<<endl;}

record.close();

reading.close();

}

void AddData(){

// временные переменные

string \_surname;

string \_name;

string \_patronymic;

int stock;

int stockSerial[99];

int stockNumber[99];

int stockValue[99];

string DateOfBuy[99];

int pasport;

int n = AmountOfData() + 1;

// открываем буферный файл

ofstream record("Buffer.txt",ios::app);

ofstream record\_("Buffer.txt", ios::out | ios::in);

fstream \_buf("Buffer.txt",ios::out);

if(\_buf.peek() == EOF){

record << n << endl;

}

if(record){

cout << "Введите фамилию: ";

cin >> \_surname;

record << \_surname << endl;

cout << "Введите имя: ";

cin >> \_name;

record << \_name << endl;

cout << "Введите отчество: ";

cin >> \_patronymic;

record << \_patronymic << endl;

cout << "Введите количество акций: ";

cin >> stock;

record << stock << endl;

cout << "Введите номер и серию паспорта: ";

cin >> pasport;

record << pasport << endl;

for(int z = 0; z < stock;z++){

cout << "Введите серию " << z+1 <<" акции: ";

cin >> stockSerial[z];

record << stockSerial[z] << endl;

cout << "Введите номер " << z+1 <<" акции: ";

cin >> stockNumber[z];

record << stockNumber[z] << endl;

cout << "Введите стоимость " << z+1 <<" акции: ";

cin >> stockValue[z];

record << stockValue[z] << endl;

cout << "Введите дату покупки " << z+1 <<" акции: ";

cin >> DateOfBuy[z];

record << DateOfBuy[z] << endl;

}

}else{cout << "Ошибка открытия файла" << endl;}

record.close();

record\_.close();

\_buf.close();

}

void DataChange(){

Copy();

ifstream reading("Buffer\_.txt");

ofstream record("Buffer.txt",ios::out);

if(reading){

if(record){

// временные переменные

string \_surname;

string \_name;

string \_patronymic;

int stock;

int stockSerial[99];

int stockNumber[99];

int stockValue[99];

string DateOfBuy[99];

int pasport;

int n, \_n;

// читаем данные из буфера в новый буфер

reading >> n;

cout << "Выберите номер изменяемого элемента от 1 до " << n\ << ":";

cin >> \_n;

\_n--;

record << n << endl;

if(\_n >= 0 and \_n < n){

for(int i = 0; i < n;i++){

if(i != \_n){

reading >> \_surname;

record << \_surname << endl;

reading >> \_name;

record << \_name << endl;

reading >> \_patronymic;

record << \_patronymic << endl;

reading >> stock;

record << stock << endl;

reading >> pasport;

record << pasport << endl;

for(int z = 0; z < stock;z++){

reading >> stockSerial[z];

record << stockSerial[z] << endl;

reading >> stockNumber[z];

record << stockNumber[z] << endl;

reading >> stockValue[z];

record << stockValue[z] << endl;

reading >> DateOfBuy[z];

record << DateOfBuy[z] << endl;

}

}else{

cout << "Введите фамилию: ";

cin >> \_surname;

record << \_surname << endl;

cout << "Введите имя: ";

cin >> \_name;

record << \_name << endl;

cout << "Введите отчество: ";

cin >> \_patronymic;

record << \_patronymic << endl;

cout << "Введите количество акций: ";

cin >> stock;

record << stock << endl;

cout << "Введите номер и серию паспорта: ";

cin >> pasport;

record << pasport << endl;

for(int z = 0; z < stock;z++){

cout << "Введите серию " <<z+1<<" акции: ";

cin >> stockSerial[z];

record << stockSerial[z] <<endl;

cout << "Введите номер" << z+1 <<" акции: ";

cin >> stockNumber[z];

record << stockNumber[z] <<endl;

cout <<"Введите стоимость"<<z+1<<"акции: ";

cin >> stockValue[z];

record << stockValue[z] <<endl;

cout << "Введите дату покупки акции № "\ <<z+1<< ": ";

cin >> DateOfBuy[z];

record << DateOfBuy[z] <<endl;

}

}

}

cout << "Данные изменены" << endl;

}else{cout <<"Номер введен неверно!" << endl;}

}else{cout<<"Ошибка открытия файла" << endl;}

}else{cout << "Ошибка открытия буферного файла"<< endl;}

record.close();

reading.close();

remove("Buffer\_.txt");

}

void DeleteData(){

Copy();

ifstream reading("Buffer\_.txt");

ofstream record("Buffer.txt",ios::out);

if(reading){

if(record){

// временные переменные

string \_surname;

string \_name;

string \_patronymic;

int stock;

int stockSerial[99];

int stockNumber[99];

int stockValue[99];

string DateOfBuy[99];

int pasport;

int n, \_n;

reading >> n;

int b = n -1;

cout << "Выберите номер изменяемого элемента от 1 до " << n\ << ":";

cin >> \_n;

\_n--;

record << b << endl;

if(\_n >= 0 and \_n < n){

for(int i = 0; i < n;i++){

if(i != \_n){

reading >> \_surname;

record << \_surname << endl;

reading >> \_name;

record << \_name << endl;

reading >> \_patronymic;

record << \_patronymic << endl;

reading >> stock;

record << stock << endl;

reading >> pasport;

record << pasport << endl;

for(int z = 0; z < stock;z++){

reading >> stockSerial[z];

record << stockSerial[z] << endl;

reading >> stockNumber[z];

record << stockNumber[z] << endl;

reading >> stockValue[z];

record << stockValue[z] << endl;

reading >> DateOfBuy[z];

record << DateOfBuy[z] << endl;

}

}else{

reading >> \_surname;

reading >> \_name;

reading >> \_patronymic;

reading >> stock;

reading >> pasport;

for(int z = 0; z < stock;z++){

reading >> stockSerial[z];

reading >> stockNumber[z];

reading >> stockValue[z];

reading >> DateOfBuy[z];

}

}

cout << "Данные удалены" << endl;

}

}else{cout <<"Номер введен неверно!" << endl;}

}else{cout<<"Ошибка открытия файла" << endl;}

}else{cout << "Ошибка открытия буферного файла"<< endl;}

record.close();

reading.close();

remove("Buffer\_.txt");

}

void SaveData(string FileName){

cout << "Введите название файла в который вы хотите сохранить”\ ”данные (с расширением файла): ";

cin >> FileName;

ifstream reading("Buffer.txt");

ofstream record(FileName,ios::out);

if(reading){

if(record){

// временные переменные

string \_surname;

string \_name;

string \_patronymic;

int stock;

int stockSerial[99];

int stockNumber[99];

int stockValue[99];

string DateOfBuy[99];

int pasport;

int n;

reading >> n;

record << n << endl;

for(int i = 0;i < n;i++){

reading >> \_surname;

record << \_surname << endl;

reading >> \_name;

record << \_name << endl;

reading >> \_patronymic;

record << \_patronymic << endl;

reading >> stock;

record << stock << endl;

reading >> pasport;

record << pasport << endl;

for(int z = 0; z < stock;z++){

reading >> stockSerial[z];

record << stockSerial[z] << endl;

reading >> stockNumber[z];

record << stockNumber[z] << endl;

reading >> stockValue[z];

record << stockValue[z] << endl;

reading >> DateOfBuy[z];

record << DateOfBuy[z] << endl;

}

}

cout << "Данные сохранены в файле" << FileName << endl;

}else{cout <<"Ошибка открытия буферного файла" << endl;}

}else{cout << "Ошибка открытия файла" << endl;}

record.close();

reading.close();

}

bool DataCleaning(){

bool clear = false;

fstream \_buf("Buffer.txt",ios::out);

if(!\_buf){

cout << "Ошибка открытия буферного файла" << endl;

}

\_buf.clear();

if(\_buf.peek() == EOF){

clear = true;

}else{

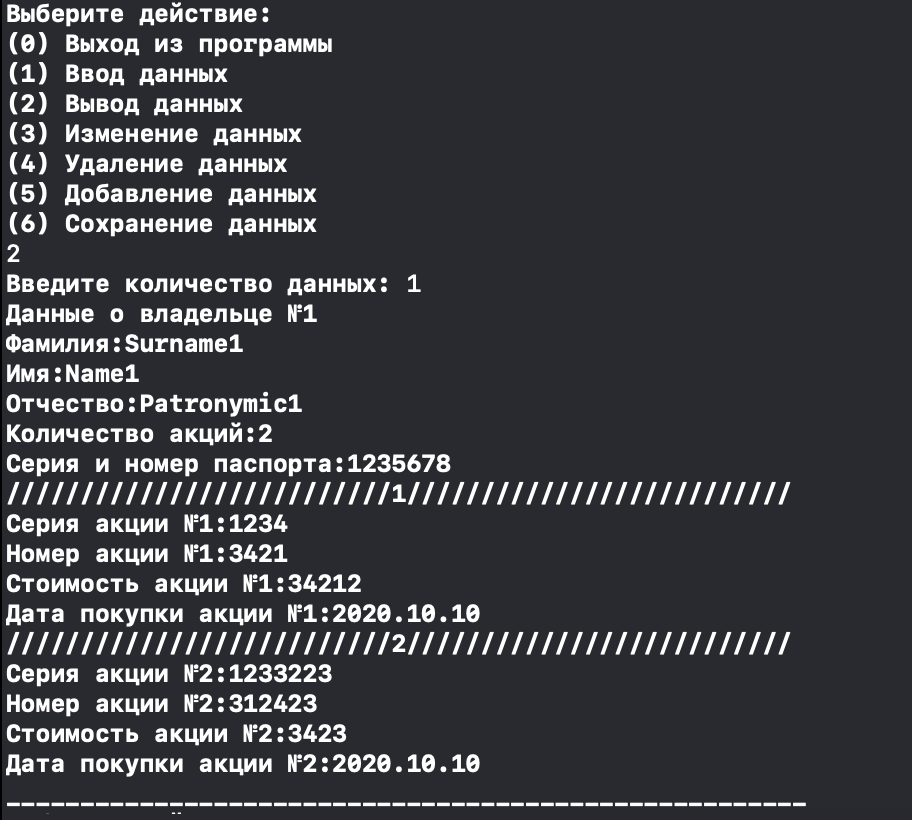
clear = false;

}

\_buf.close();

return clear;

}

**Результат работы программы**

