СОДЕРЖАНИЕ

1. ЗАДАНИЕ 3

2. ОПИСАНИЕ ПРИМЕНЕНИЯ 3

2.1. Запуск программы 3

2.2. Входные данные 3

2.3. Выходные данные 3

2.4. Сообщения программы 4

3. ОПИСАНИЕ ПРОГРАММЫ 4

3.1. Метод решения задачи 4

3.2. Структура программы 4

3.3. Описание функций 5

3.3.1. main – главная функция 5

3.3.2. Dobavlenie - добавляет информацию, введенную с

клавиатуры, в конец файла**.** 7

3.3.3.Vivod - выводит информацию о товарах. 8

ЛИТЕРАТУРА 12

Приложение 1. Текст программы 13

Приложение 2. Пример входного файла avto.txt 18

Приложение 3. Результаты тестирования программы 18

**1. ЗАДАНИЕ**

Дан файл, содержащий информацию о наличии товаров на складе.  
 Структура записи файла: шифр товара, наименование, количество, цена. В  
 файле записи не упорядочены, количество записей не более 50.  
 Написать программу, которая  
 - добавляет новую запись, вводимую с клавиатуры, в конец файла;  
 - печатает данные в виде таблицы в порядке возрастания шифров  
 товаров.

**2. ОПИСАНИЕ ПРИМЕНЕНИЯ**

**2.1 Запуск программы**

Запуск программы производится в среде программирования CodeBlocks

**2.2 Входные данные**

Входные данные находятся в файле “tovar.txt”. Число строк в файле - произвольное (но не более 50). Каждая строка содержит шифр товара(8 символов), наименование товара(7 символов), количество товара(5 символов), количество товара(3 символа). Пример строки файла:

aaa111, мидии 1200 20.

| 8 | 7 | 5 | 3 |

Пример входного файла приведен в приложении 2.

По запросу программы с клавиатуры необходимо ввести номер пункта меню (см. раздел 2.3).

**2.3. Выходные данные**

Программа выводит на экран меню:

===========================================================

**Выберите номер пункта меню:**

**1 - Добавление новой записи в конец файла**

**2 - Печать данных**

**3 - Выход**

При выборе пункта 1 нужно ввести шифр, наименование, количество и цену товара, затем эта запись занесется в конец файла.

При отсутствии файла выводится сообщение:

Файл не может быть открыт!

Примеры выходных данных для остальных пунктов меню см. в приложении 3.

**2.4. Сообщения программы**

Ниже приводится перечень возможных сообщений программы:

**1.Файл не может быть открыт!**

**2.Выберите номер пункта меню:**

**3.Нужно вводить номер пункта от 1 до 3**

**4.Для продолжения нажмите любую клавишу**

**5.Файл пуст!**

**6.Запись в файл завершена**

**7.Вывод завершен**

**8.№ Шифр Н-ие К-во Цена**

**9.Информация о товарах отсутствует!**

**3. ОПИСАНИЕ ПРОГРАММЫ**

**3.1. Метод решения задачи**

Задачу можно разбить на две отдельные подзадачи:

1. Добавление в конец файла информации о новом товаре.

2. Вывод всех товаров.

Каждая подзадача решается методом линейного поиска (последовательного просмотра элементов таблицы).

Чтобы пользователь мог выбирать, какие подзадачи решать и в каком порядке, программа выводит на экран меню (см. раздел 2.3).

**3.2. Структура программы**

Vivod

Dobavlenie

main

Рис.1. Функциональная структура программы

Программа состоит из трех функций: главной функции main и двух подпрограмм.

Dobavlenie - добавляет информацию, введенную с клавиатуры, в конец файла.

Vivod - выводит информацию о товарах.

**3.3. Описание функций**

**3.3.1. main – главная функция**

Заголовок функции:

int main ()

Значение функции:

0 – в случае успешного завершения;

1 – если входной файл не найден.

Рабочие данные:

f – объект класса ifstream, с помощью которого получают доступ к указанному файлу;

n – номер пункта меню.

n

чтение номера пункта меню n

вывод меню

f=NULL

ifstream f(”tovar.txt”)

да нет

вывод “Файл не может быть открыт!”

Dobavlenie

'1'

Vivod

'2'

n!='3'

да

нет

закрытие файла

Рис.1 Блок-схема функции main()

**3.3.2.Dobavlenie - добавляет информацию, введенную с клавиатуры, в конец файла.**

Заголовок функции: Структура записи файла:

void Dobavlenie(int kolvozn, tovar tz[])

shifr - шифр

name - наименование товара

colvo - количество товара

price - цена товара

Входные данные:

f - объект класса ifstream, с помощью которого получают доступ к указанному файлу

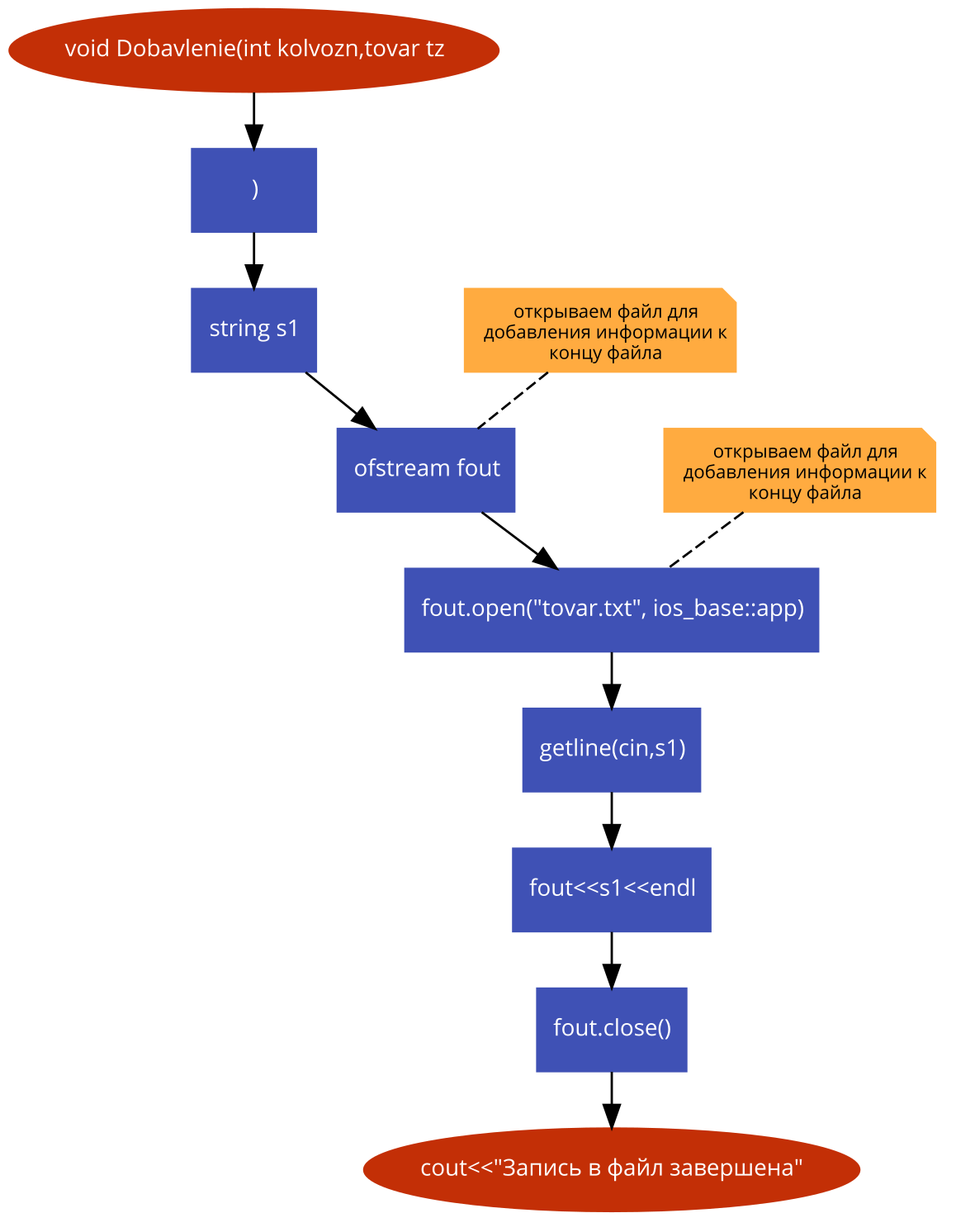
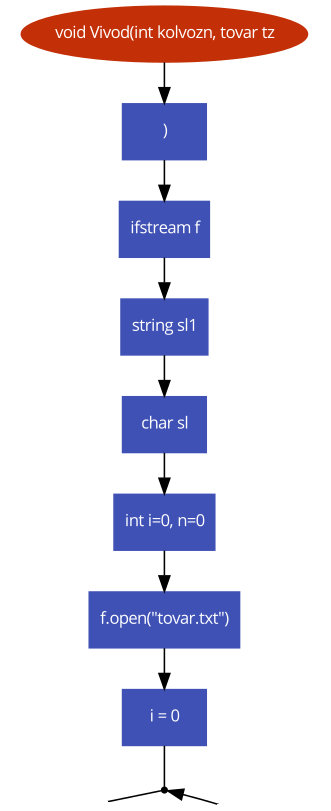


Рис.2. Схема функции добавления записи в конец файла.

**3.3.3. Vivod - выводит информацию о товарах.**

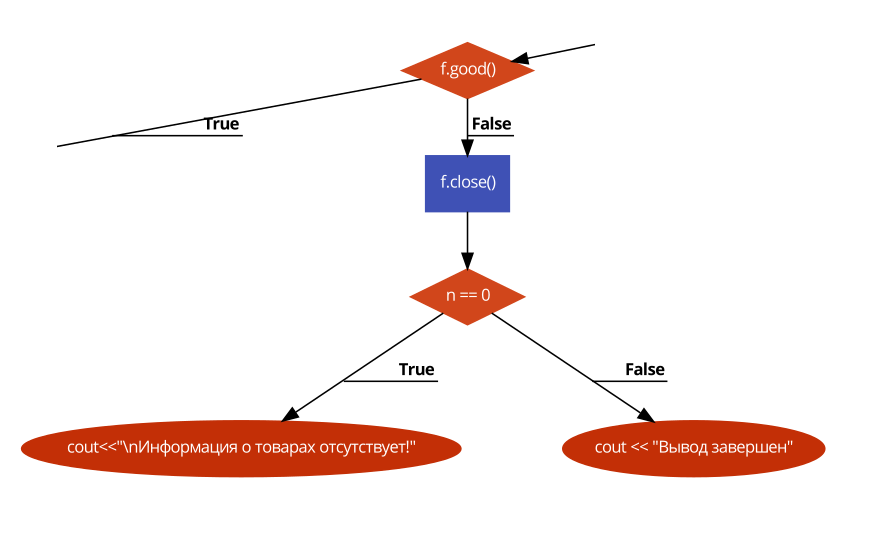
Заголовок функции:

void Vivod(int kolvozn, tovar tz[])



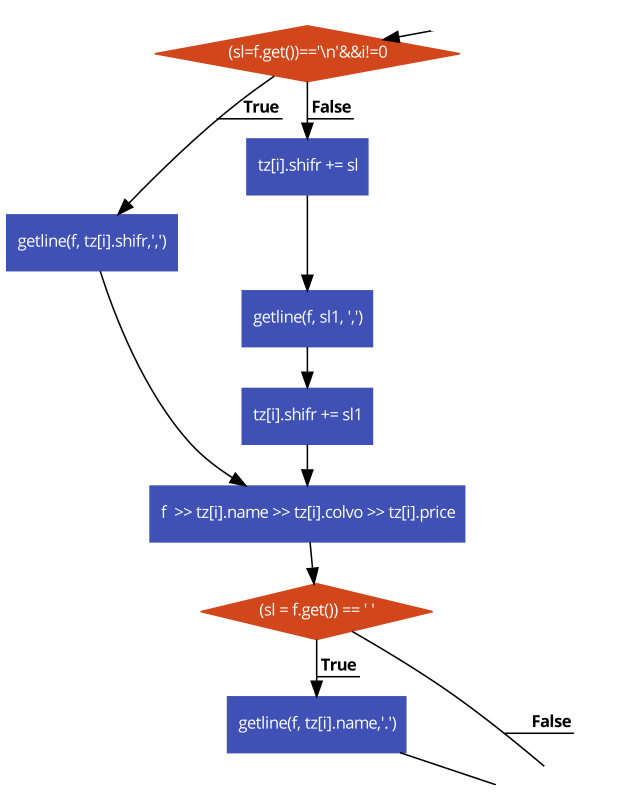
2

1



3

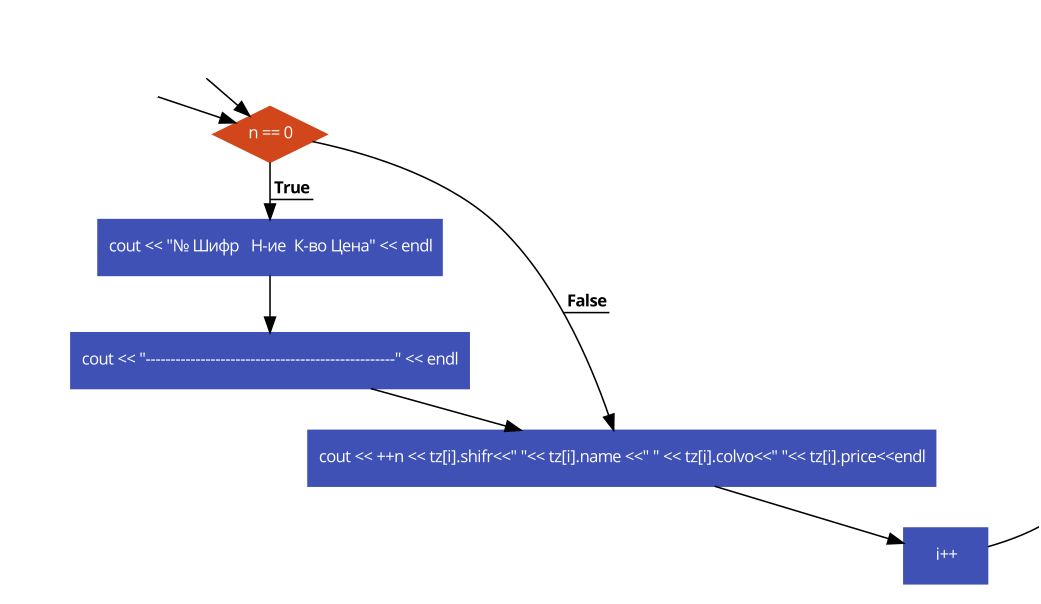
1



5

4

3



5

2

4

Рис.3. Блок-схема функции вывода информации о товарах.

**Литература.**

1. Пример кода программы.

2. Пример титульного листа.

3. Пример отчета по курсовой работе.

4. Задания к курс. работе.

**Приложение 1. Текст программы**

#include <fstream>

#include <string>

#include <iostream>

#include <sstream>

#include <windows.h>

using namespace std;

struct tovar /\* структура записи входного файла \*/

{

string shifr; /\* шифр \*/

string name; /\* наименование \*/

string colvo; /\* кол-во товара \*/

string price; /\* цена \*/

};

/\* прототипы функций \*/

void Dobavlenie(int kolvozn, tovar tz[]);

void Vivod(int kolvozn, tovar tz[]);

/\*----------------------\*/

/\* главная функция \*/

/\*----------------------\*/

int main()

{

ifstream f; /\* ссылка на входной файл \*/

char n; /\* номер пункта меню \*/

int i;

char sl;

string sl1;

SetConsoleCP(1251);// установка кодовой страницы win-cp 1251 в поток ввода

SetConsoleOutputCP(1251); // установка кодовой страницы win-cp 1251 в поток вывода

f.open("tovar.txt");

if (!f.is\_open()) // если файл не открыт

{

cout << "Файл не может быть открыт!\n";

cin.get();

return 1;

}

else if (!f.eof())

{

tovar tz[88];

for (i = 0; f.good();i++)

{

if ((sl=f.get())=='\n'&&i!=0)

getline(f, tz[i].shifr,',');

else

{

tz[i].shifr += sl;

getline(f, sl1, ',');

tz[i].shifr += sl1;

}

f >> tz[i].name >> tz[i].colvo >> tz[i].price;

if ((sl = f.get()) == ' ')

getline(f, tz[i].name,'.');

cout << tz[i].shifr<<" "<< tz[i].name <<" " << tz[i].colvo<<" "<< tz[i].price<<endl;

}

int kolvozn = i - 1;

do

{

cout << "========================================================"<<endl;

cout << "Выберите номер пункта меню:"<<endl;

cout << "1 - Добавление новой записи в конец файла" << endl;

cout << "2 - Печать данных"<<endl;

cout << "3 - Выход"<<endl;

cout << "------------------------------------------------------"<<endl;

n=cin.get();

cin.sync();

switch (n)

{

case '1': Dobavlenie(kolvozn,tz); break;

case '2': Vivod(kolvozn,tz); break;

case '3': break;

default: cout << "\nНужно вводить номер пункта от 1 до 3";

}

if (n != '3')

{

cout << "\nДля продолжения нажмите любую клавишу";

cin.get();

}

} while (n != '3');

f.close();

}

else

cout << "Файл пуст!\n";

//cin.get();

return 0;

}

void Dobavlenie(int kolvozn,tovar tz[])

{

string s1;

ofstream fout; // открываем файл для добавления информации к концу файла

fout.open("tovar.txt", ios\_base::app); // открываем файл для добавления информации к концу файла

getline(cin,s1);

fout<<endl<<s1;

fout.close();

cout<<"Запись в файл завершена";

}

void Vivod(int kolvozn, tovar tz[])

{

ifstream f;

string sl1;

char sl;

int i=0, n=0;

f.open("tovar.txt");

for (i = 0; f.good();i++)

{

if ((sl=f.get())=='\n'&&i!=0)

getline(f, tz[i].shifr,',');

else

{

tz[i].shifr += sl;

getline(f, sl1, ',');

tz[i].shifr += sl1;

}

f >> tz[i].name >> tz[i].colvo >> tz[i].price;

if ((sl = f.get()) == ' ')

getline(f, tz[i].name,'.');

if (n == 0)

{

cout << "№ Шифр Н-ие К-во Цена" << endl;

cout << "--------------------------------------------------" << endl;

}

cout << ++n << tz[i].shifr<<" "<< tz[i].name <<" " << tz[i].colvo<<" "<< tz[i].price<<endl;

}

f.close();

if (n == 0) cout<<"\nИнформация о товарах отсутствует!";

else cout << "Вывод завершен";

}

**Приложение 2. Пример входного файла avto.txt**

aaa111, мидии 1200 20.

aaa112, батон 2300 25.

aaa113, лаваш 3400 30.

bbb221, масло 4500 40.

bbb222, кефир 5600 45.

ccc331, сахар 6700 50.

fds213, чайок 3421 23.

**Приложение 3. Результаты тестирования программы**

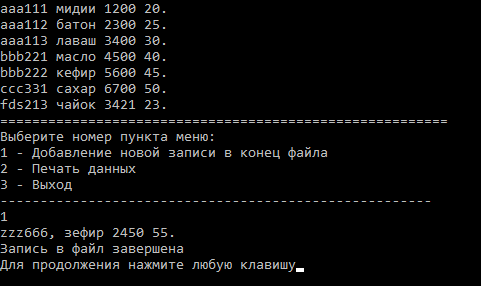
**Тест 1.** Входного файла нет в текущем каталоге.

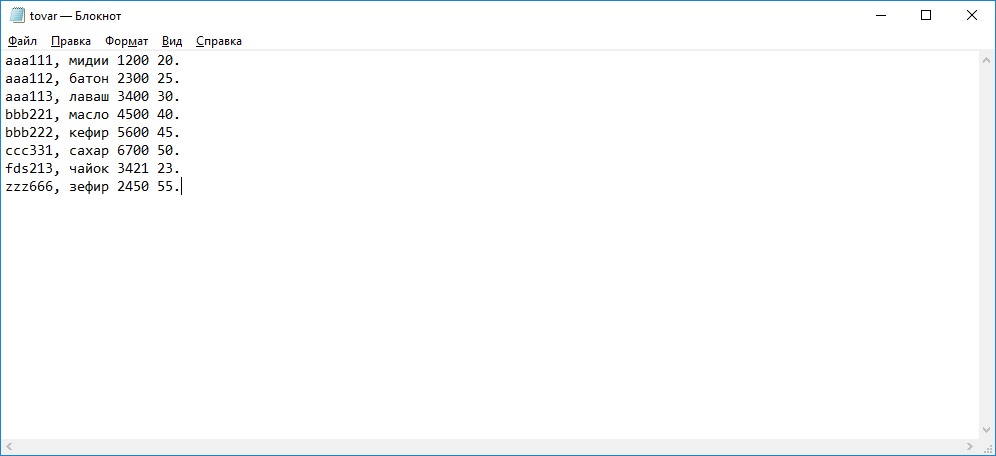
Результат:



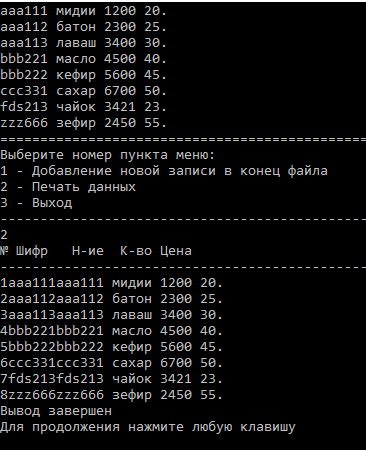
В следующих тестах используется файл из приложения 2.

**Тест 2.**

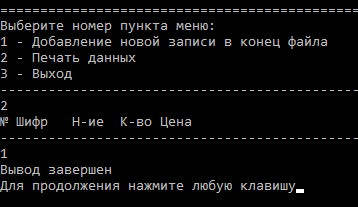
****

****

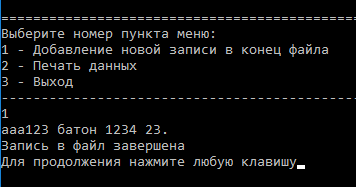
**Тест 3.**

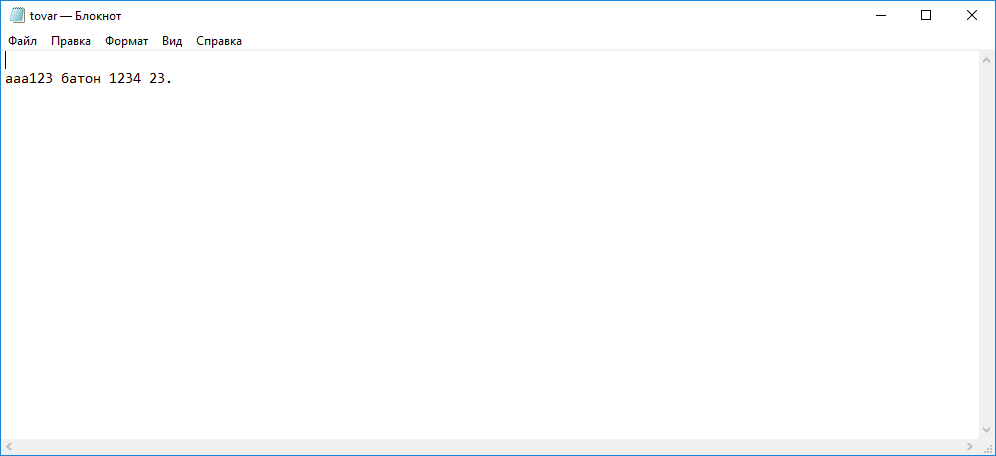
****

**Тест 4.**

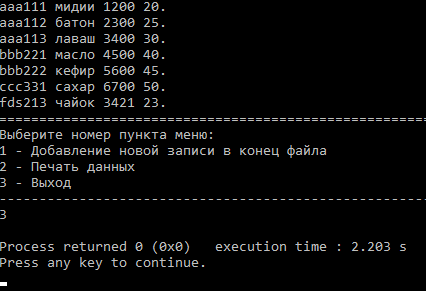
****

**Тест 5.**

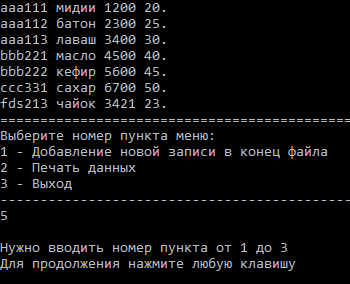
****

****

**Тест 6.**

****

**Тест 7.**

****