Министерство образования Российской Федерации

Пензенский государственный университет

Кафедра «Вычислительная техника»

**ПОЯСНИТЕЛЬНАЯ ЗАПИСКА**

к курсовой работе

по курсу «Программирование»

на тему «Разработка программы сложной структуры методом многомодульного программирования»

наименование программы «База данных. Список компьютерных игр»

**Выполнил студент группы 21ВВ4:**

Калыева С.К.

**Приняли:**

Слепцов Н.В., Патунин Д.В.

Пенза 2022

**ПЕНЗЕНСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ**Факультет Вычислительной техники

Кафедра “Вычислительная техника”

“УТВЕРЖДАЮ”

Зав. кафедрой ВТ

Митрохин М.А.

«\_\_\_\_»\_\_\_\_\_\_\_\_\_\_20\_\_

**ЗАДАНИЕ**

**на курсовое проектирование по курсу**

**Программирование**

Студенту\_ ***Калыевой С.К.*** Группа\_ ***21ВВ4***\_\_

Тема\_проекта: ***Разработка программы сложной структуры методом нисходящего программирования***.

**Исходные данные(технические требования)на проектирование**

**Наименование программы**: ***База данных***\_\_***Список компьютерных игр***\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_.

**Задание**: Программа должна выполнять следующие функции:

* Создание новой базы данных. Добавление записей
* Удаление записей. Редактирование записей
* Поиск и сортировка записей. Фильтрация записей по определенному критерию.
* Сохранение текущей базы данных в файл. Загрузка сохраненных баз данных из файла.

**Обязательные требования к программе**:

1. Многомодульность.
2. Использование сложных типов данных – списков, структур, файлов.
3. Режим работы видеосистемы – текстовый/графический.
4. Устройство ввода информации – клавиатура, мышь.
5. Пользовательский интерфейс должен быть построен на основе меню и панели инструментов.
6. Наличие заставки.
7. Операционная система – MS Windows
8. Система программирования – Си/Си++  на базе Среды  Microsoft  Visual  Studio 2015.
9. Язык программирования – Си и Ассемблер

**Расчетная часть**

*Разработк программы* \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**Графическая часть**

*Схема данных, схема программы, схема работы системы, иерархическая структура программы, схема взаимодействия программы.* \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**Экспериментальная часть**

*Отладка программы* \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**Срок выполнения проекта по разделам**

1. *В соответствии с графиком выполнения курсового проекта*\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

2.\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ 3.\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

4.\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

5.\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Дата выдачи задания “\_21\_”\_марта\_2020 г.\_\_\_\_\_\_\_\_\_

Дата защиты проекта “\_\_\_\_”\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**Руководитель\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

**Задание получил** “\_\_\_\_\_\_”\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_20\_\_\_г.

**Студент\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

Содержание

[Введение 4](#_Toc42213709)

[1. Постановка задачи. 4](#_Toc42213710)

[2. Выбор решения. 5](#_Toc42213711)

[2.1. Определение необходимых модулей программы. 5](#_Toc42213712)

[2.2 Определение структуры базы данных 6](#_Toc42213713)

[3. Описание разработки программы. 7](#_Toc42213714)

[4. Отладка и тестирование. 10](#_Toc42213715)

[5. Описание программы. 10](#_Toc42213716)

[6. Руководство пользователя 12](#_Toc42213717)

[Заключение 17](#_Toc42213718)

[Список используемых источников 18](#_Toc42213719)

[Приложение 19](#_Toc42213720)

[Листинг 20](#_Toc42213721)

# Введение

База данных – это совокупность связанных данных, организованных по определенным правилам, предусматривающим общие принципы описания, хранения и манипулирования, независимая от других программ.

Главной задачей моей курсовой работы является разработка базы данных «список компьютерных игр», которая предполагает создание программы на языке Си с использованием класса статических массивов структур.

Структуры.

Структура – это составной тип данных, который получается путем объединения компонентов, принадлежащих к другим типам данных. Структура предназначена для объединения нескольких переменных в один тип данных.

# 1. Постановка задачи.

Необходимо разработать программу «База данных. Список компьютерных игр». В данной программе информация структурирована по следующим полям: «Название», «Разработчик», «Жанр», «Количество проданных копий», «Год выпуска». Для реализации данной программы необходимо определить структуру базы данных, размер, размер каждой записи базы данных.

Программа должна позволять пользователю осуществлять обработку данных, для этого в программе должны присутствовать следующие функции:

* Создание новой базы данных
* Добавление записей
* Удаление записей
* Поиск и сортировка записей
* Редактирование записей
* Фильтрация записей по определенному критерию
* Сохранение текущей базы данных в файл.
* Чтение базы данных из файла

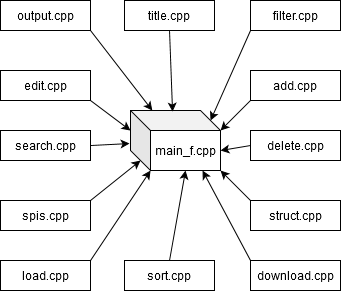
Для реализации вышеперечисленных требований необходимо разработать интуитивно понятный интерфейс пользователя, определить названия пунктов меню, определить тип интерфейса, продумать навигацию по базе данных, изучить функции для работы с файлами. Это необходимо для четкого понимания функциональности программы.

В программе должны использоваться сложные типы данных: структуры, массивы, списки, файлы. Это необходимо для структурирования базы данных, для обработки данных в коде программы, для просмотра базы данных, с использованием файла, не используя запуск программы.Устройствами ввода являются клавиатура и мышь. Навигация по пунктам меню и данные вводятся посредством клавиатуры. Программа должна быть разработана для работы в операционной системе MS Windows.Средой разработки является Microsoft Visual Studio 2019, языкпрограммирования Си и Ассемблер.

# 2. Выбор решения.

# 2.1. Определение необходимых модулей программы.

Разработанная программа состоит из следующих модулей



**Рисунок 1 – Структура модулей программы**

Файлы: «struct.cpp», «delete.cpp», «edit.cpp», «download.cpp», «flter.cpp»,«load.cpp», «main\_f.cpp», «output.cpp», «search.cpp», «sort.cpp», «spis.cpp», «add.cpp»необходимыдлявключенияфункцийобработкиданных. Данные файлы содержат необходимые функции чтения, поиска, редактирование, добавления новых записей, создания, сортировки, фильтрации, вывода на экран базы данных и функцию сохранения результатас загрузкой из базы обработки базы данных.

Все вышеперечисленное удовлетворяет требованию модульности программы данной курсовой работы. Управление приложением осуществляется с помощью клавиатуры. При запуске программы выводится заставка-приветствие, после чего программа будет готова к дальнейшему исполнению. Программа предназначена для работы с базой данных, база данных находится в файле, файл так же можно создать самому. Открытие и сохранение выходного файла выполняется средствами данной программы. Программа дает возможность просматривать содержимое базы данных, искать данные, сортировать их. Данные из базы отображаются в программе в виде таблицы из пяти столбцов«Name», «Developer», «Genre», «Sells», «Year».

# 2.2 Определение структуры базы данных

typedefstructgame\_list

{

char nazvanie[16],

developer[16],

genre[16],

sells[16],

year[16];

game\_list\* next;

};

Поля nazvanie(«Название»), developer(«Разработчик»), genre(«Жанр»), sells(«Количество проданных копий»), year(«Годвыпуска») представляют собой символьную последовательность, имеющую ограничение в 16 символов.

Выбранная структура проста и соответствует всем требованиям программы.

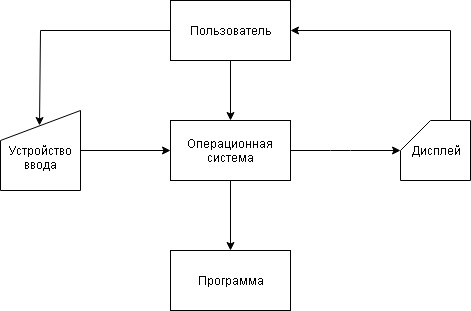


Рисунок 2. - Схема данных

# 3. Описание разработки программы.

В качестве среды программирования был выбран программный продукт MicrosoftVisualstudio 2019

Интерфейс базы данных представляет собой меню.Перемещение по пунктам меню осуществляется при помощи цифр от 1 до 11, вводимых с клавиатуры.

При создании базы данных, после ввода каждого узла динамического списка будет выводиться сообщение о прекращении создании базы данных. Прекращение создания осуществляется вводом с клавиатуры ‘`’.

При выборе пункта меню «Createdatabase», создается база данных, в навигационном меню предусмотрен пункт для обновления базы данных «Add». Данный пункт позволяет добавить новый узел в конец существующей базы данных.

Для корректировки предусмотрена функция удаления последнего узла, находящаяся в пункте «Deletethelastline» навигационного меню.

Также в программе описаны такие пункты меню, как «Filter», «Search». Первый пункт осуществляет возможность отбора данных по определенному критерию. Второй пункт дает пользователю возможность найти в базе данных необходимую информацию по определенному слову.

Пункт меню «Load» предоставляет возможность представить базу данных в текстовом формате, создавая для этого файл. Также пользователь вводит нужное ему имя файла и разрешение, после чего произойдет его создание и запись базы данных.

Пункт меню «Databaseoutput» позволяет в любой момент использования программы выбрать его и тем самым просмотреть содержимое существующей базы данных.

Пункт меню «Getfromfile» позволяет загрузить базу данных из файла в память программы.

Пункт меню «Edit» позволяет вносить изменения в уже существующую базу данных.

Пункт меню «Sort» позволяет сортировать базу данных по Названию или году выпуска.

После завершения работы с базой данных для выхода из программы существует пункт меню «Exit». После выполнения данного пункта существующая база данных будет удалена из памяти программы.

Разработка программы началась с файла **spis.cpp**, в котором первой была написана основная функция для базы данных – это функция создания базы данныхspisok. Для быстрого поиска данных, находящихся в базе данных, была создана функция **search**(), которая осуществляет поиск по всем полям («Название», «Разработчик», «Жанр», «Количество проданных копий», «Год выпуска»). Затем были описаны функции для вывода созданной базы данных на экран – out(),сохранения базы данных в файл – **load**().Также для функции**del**() позволяет удалить последний узел динамического списка. Функция **output**() отвечает за корректный вывод базы данных на экран. Функция**filter**() позволяет отфильтровать базу данных по введенному фильтру по всем возможным полям. Функция **edit**() позволяет редактировать базу данных. Функция **sort**()позволяет сортировать базу данных по двум полям Название и год выпуска. Функциями информации и навигационного меню является функция **main**(). Функция **main**() является главной функцией проекта и находится в файле **main\_f.ccp**. Главной ее задачей является осуществление связи между информационными функциями и функциями, выполняющие задачи, указанные в навигационном меню и вывод навигационного меню. На экран печатается навигационное меню, после чего запрашивается ввод цифры от 1 до 11. Каждой цифре соответствует определенный пункт меню. После выбора пункта меню выбирается подпункт меню, затем вызывается нужная функция. Схема программы функции **main**() представлена на рисунке ниже.

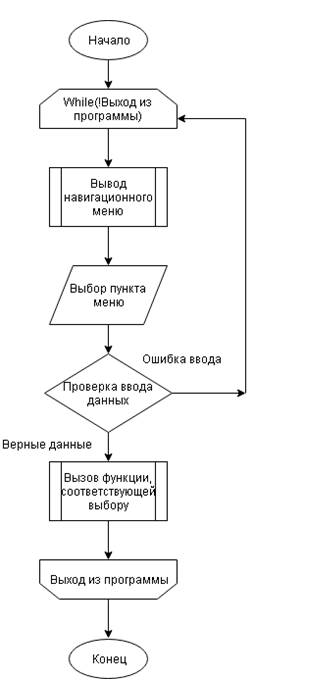


Рисунок 3. - Схема программы функции main()

В зависимости от пункта навигационного меню, после выбора может вызваться либо подменю, либо выполнение определенной функции. Алгоритмописанныхдействийпредставленниженарисунке.

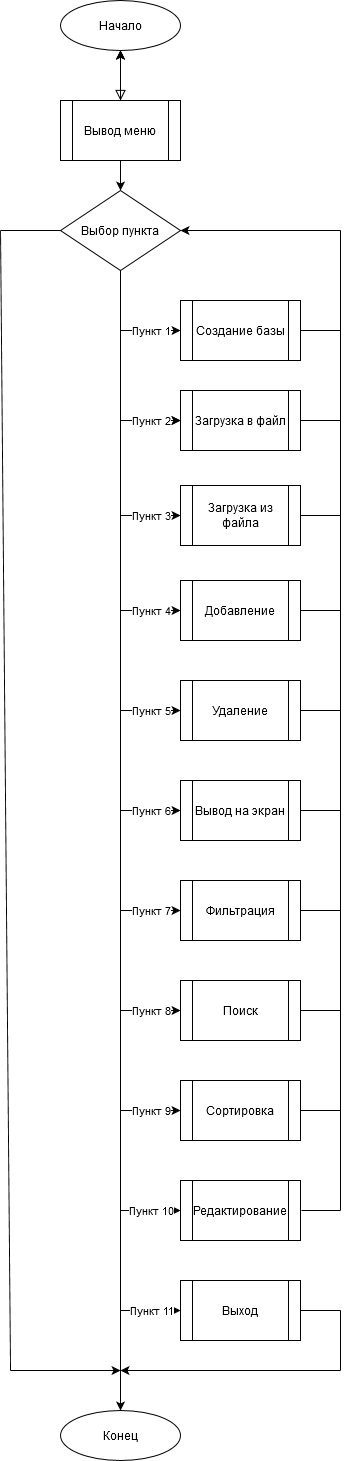


Рисунок 4. - Схеманавигационногоменюmain\_f.cpp

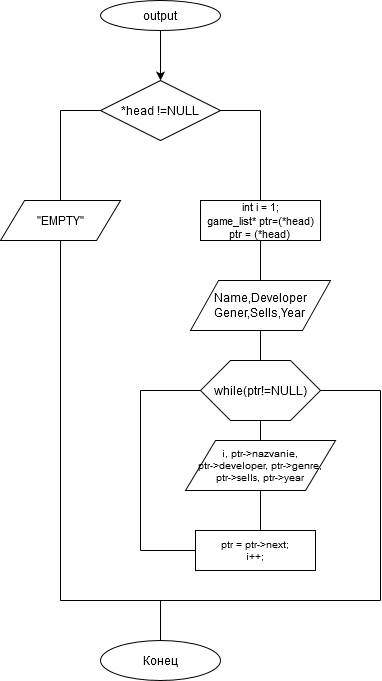
# 4. Отладка и тестирование.

ПрисозданиипроектаиспользоваласьпрограммаMicrosoftVisualStudio 2019. Онаимеетвсенеобходимыедляразработкииотладкипрограммы. Дляотладкипрограммыиспользовались: точкиостановок, анализсодержимоголокальныхиглобальныхпеременных, пошаговоевыполнениекода.

5. Описание программы.  
  
Послезапускапрограммыпоявляетсянавигационноеменю, содержащеепункты, которыеимеютопределеннуюнумерацию. Длявыборанеобходимогопунктаменюдоступенвводцифры, которойсоответствуетопределенныйпунктменю. Послевводацифрыосуществляетсявыполнениефункции, тоестьожидаетсявыполнениеопределенногодействия, либо выводится подменю, в котором нужно выбрать необходимое действие.

Структура главного меню:

|  |  |  |
| --- | --- | --- |
| № пункта | Опция | Действия программы |
| 1 | Create date base | Производится создание новой базы данных |
| 2 | Load | База загружается в файл |
| 3 | Get from file | База загружается из файла |
| 4 | Add | Добавление новых строк |
| 5 | Delete the last line | Удаление последней строки |
| 6 | Data base output | Вывод базы данных |
| 7 | Filter | Фильтрация |
| 8 | Search | Поиск записей |
| 9 | Sort | Сортировка |
| 10 | Edit | Редактирование |
| 11 | Exit | Запрос на подтверждение выхода |

Блок схема для функцииoutput(вывод базы на экран)

**Рисунок 5 – Схема программирования для функции output**

# 6. Руководство пользователя

При запуске программы у нас появляется титульный лист.

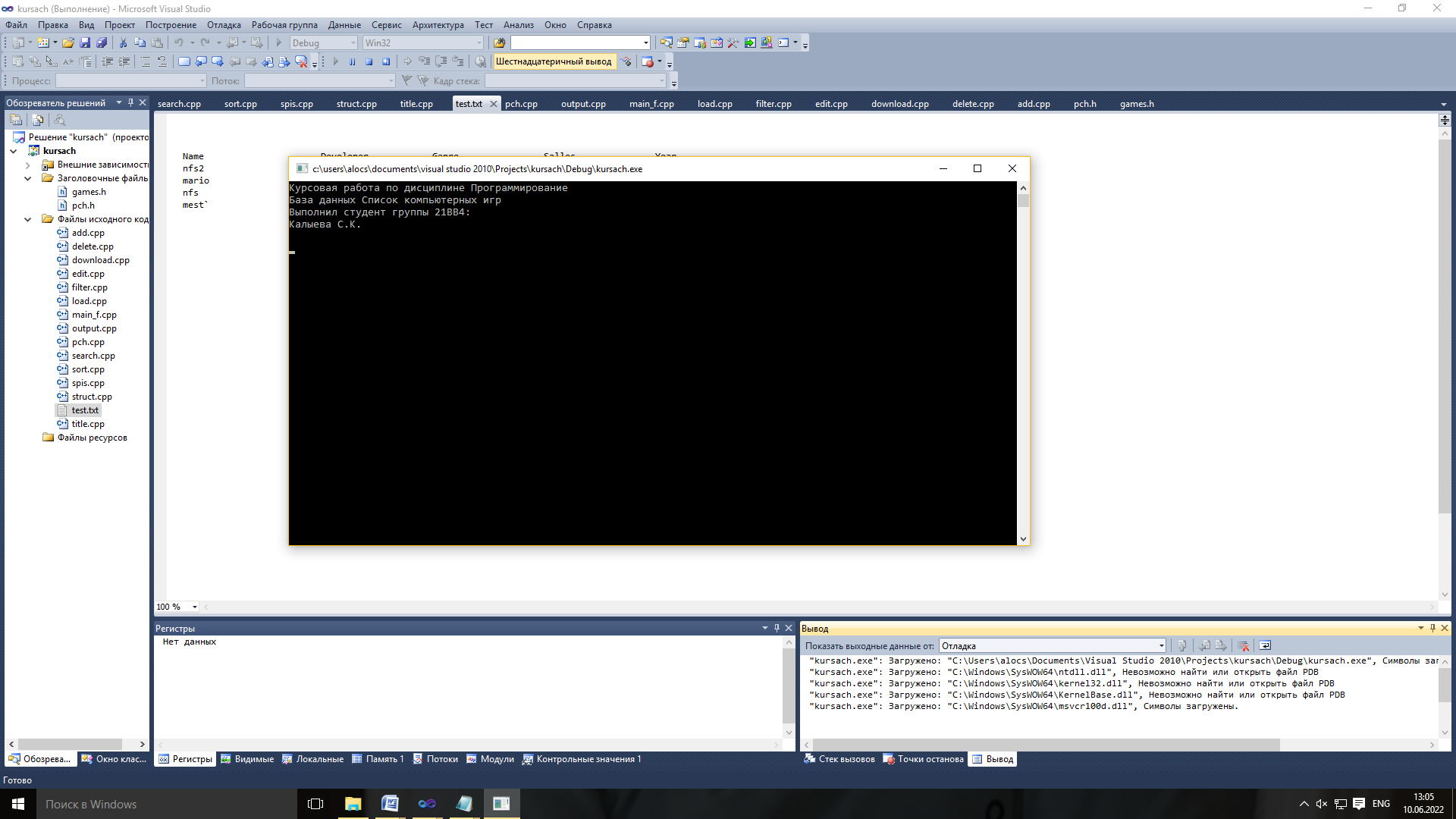


Рисунок 6. - Титульный лист

Для чтения и вывода на экран существующего списка необходимо выбрать пункт меню «Getfromfile», либо можно создать новую базу данных, выбрав «Createdatabase», и нажать клавишу Enter.

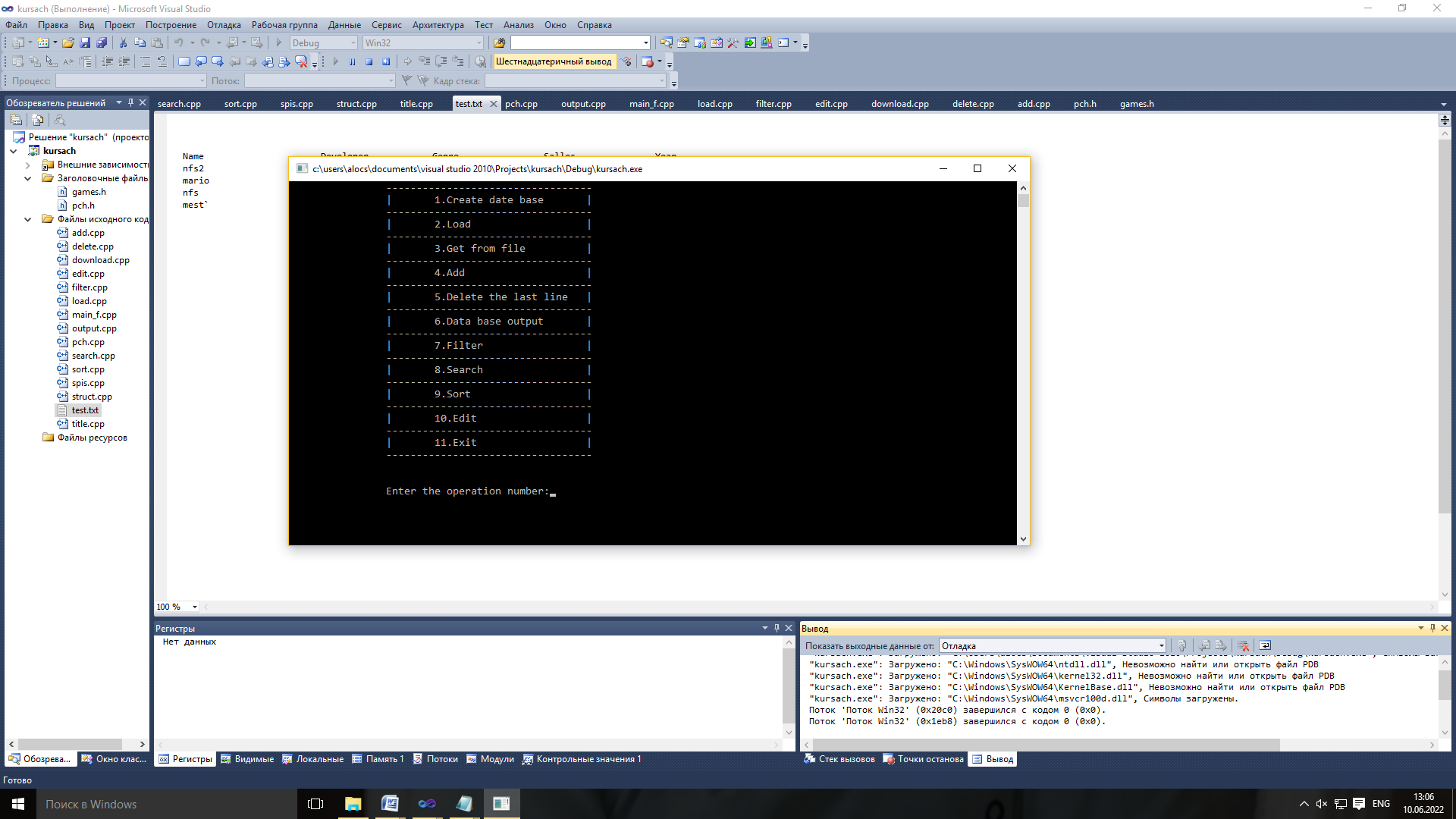


Рисунок 7. - Меню программы

После этого произойдет чтение и вывод базы данных.

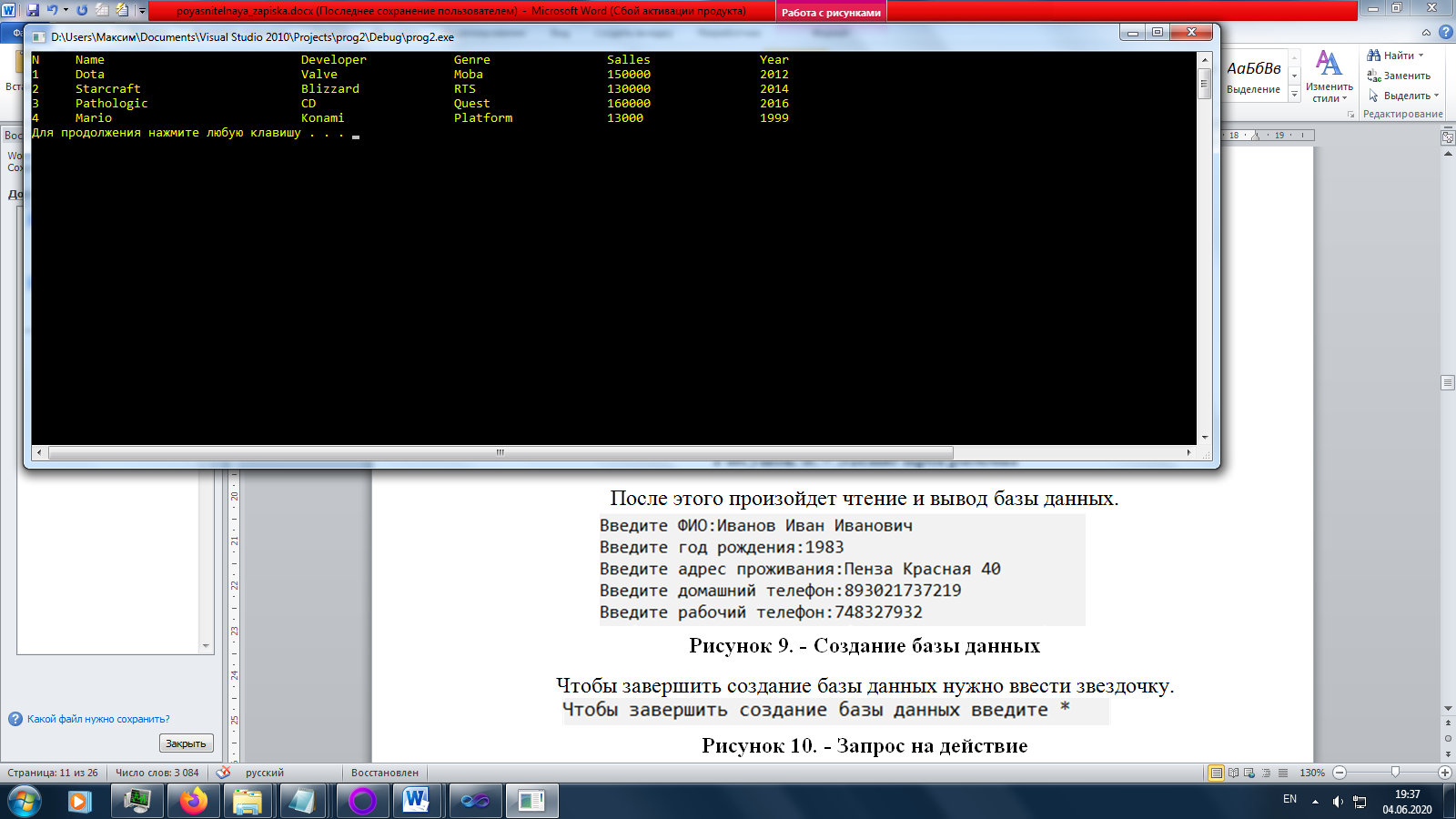


Рисунок 8. –Вывод базы данных

Для редактирования базы данных следует выбрать пункт 10.

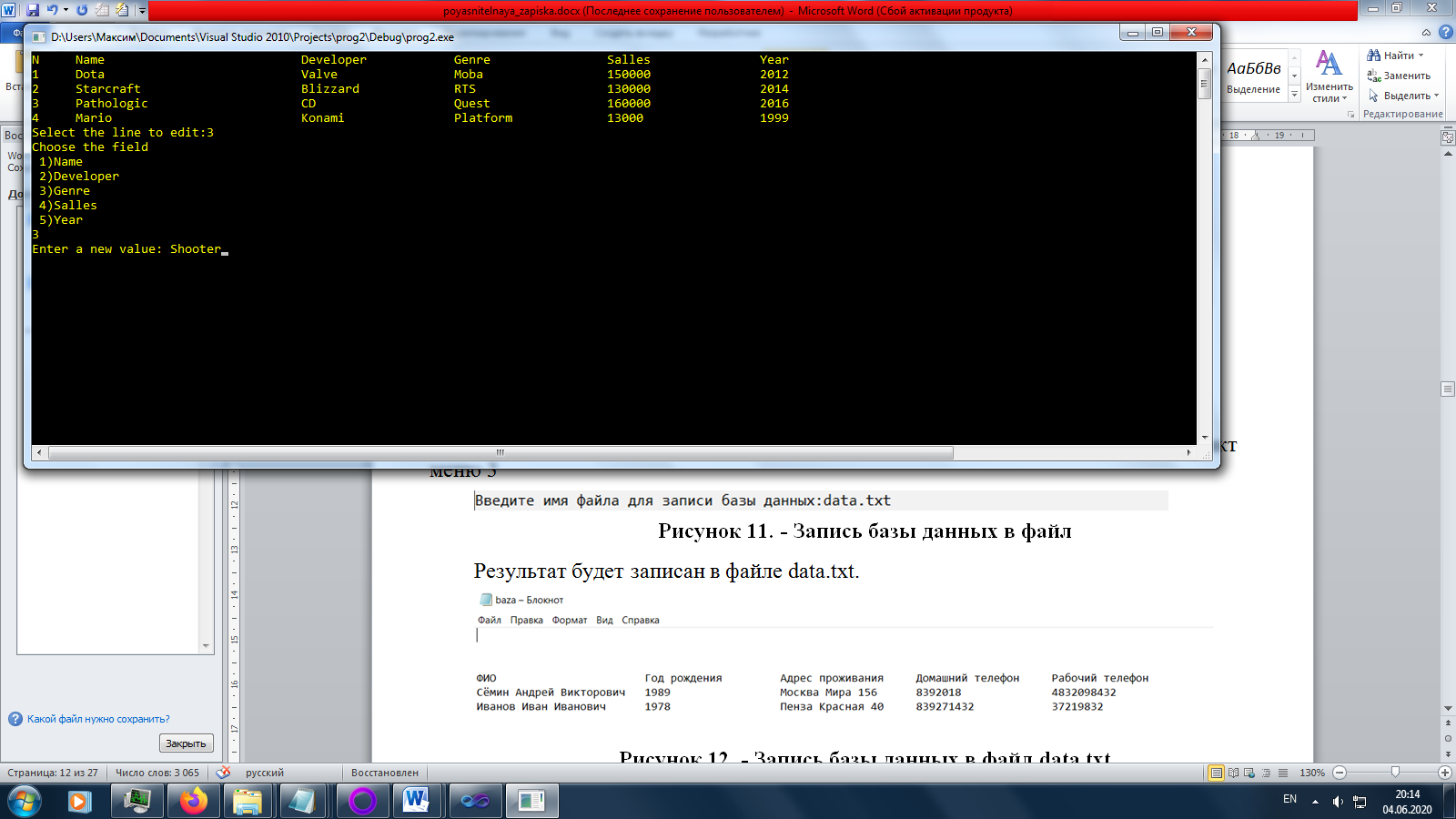


Рисунок 9. –Редактирование

После окончания ввода следует нажать клавишу Enter. В случае успешного выполнение операции программа выйдет в главное меню.

Для создания файла и записи в него созданной базы данных предназначен пункт меню 2

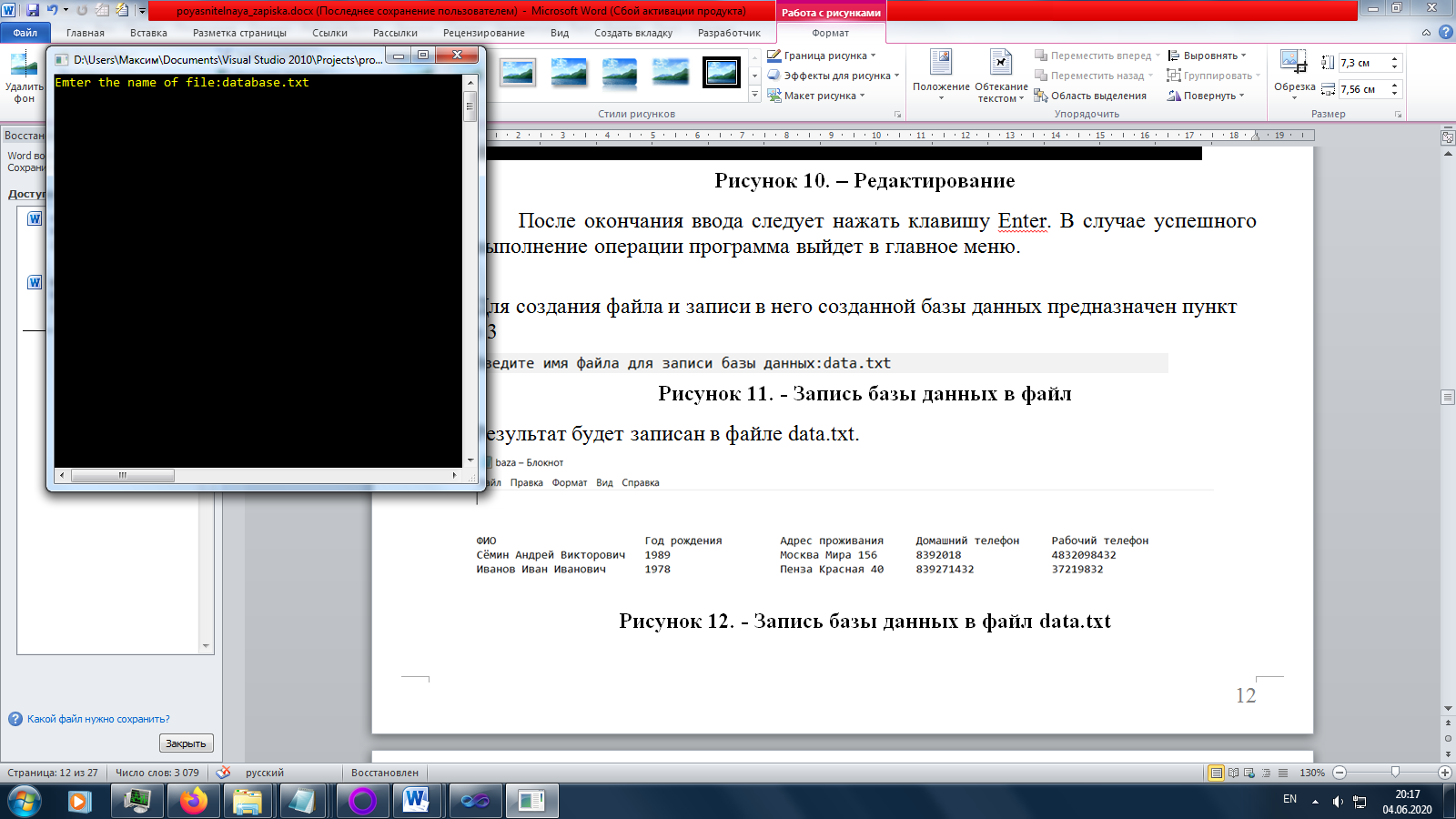


Рисунок 10. - Запись базы данных в файл

Результат будет записан в файле database.txt.

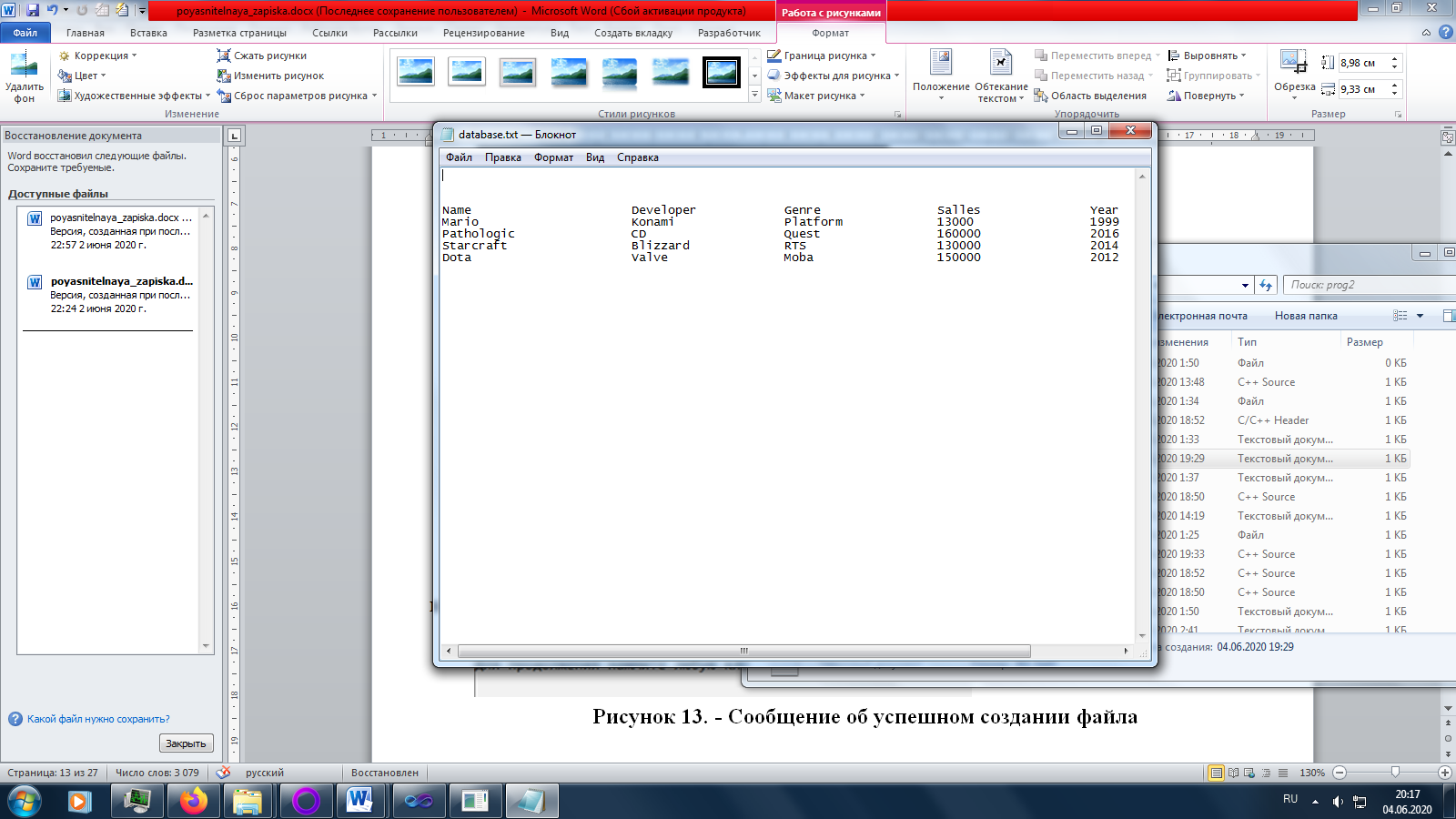


Рисунок 11. - Запись базы данных в файл database.txt

Для получения базы данных из файла нужно воспользоваться пунктом 3 и ввести название файла, из которого будет получена база данных.

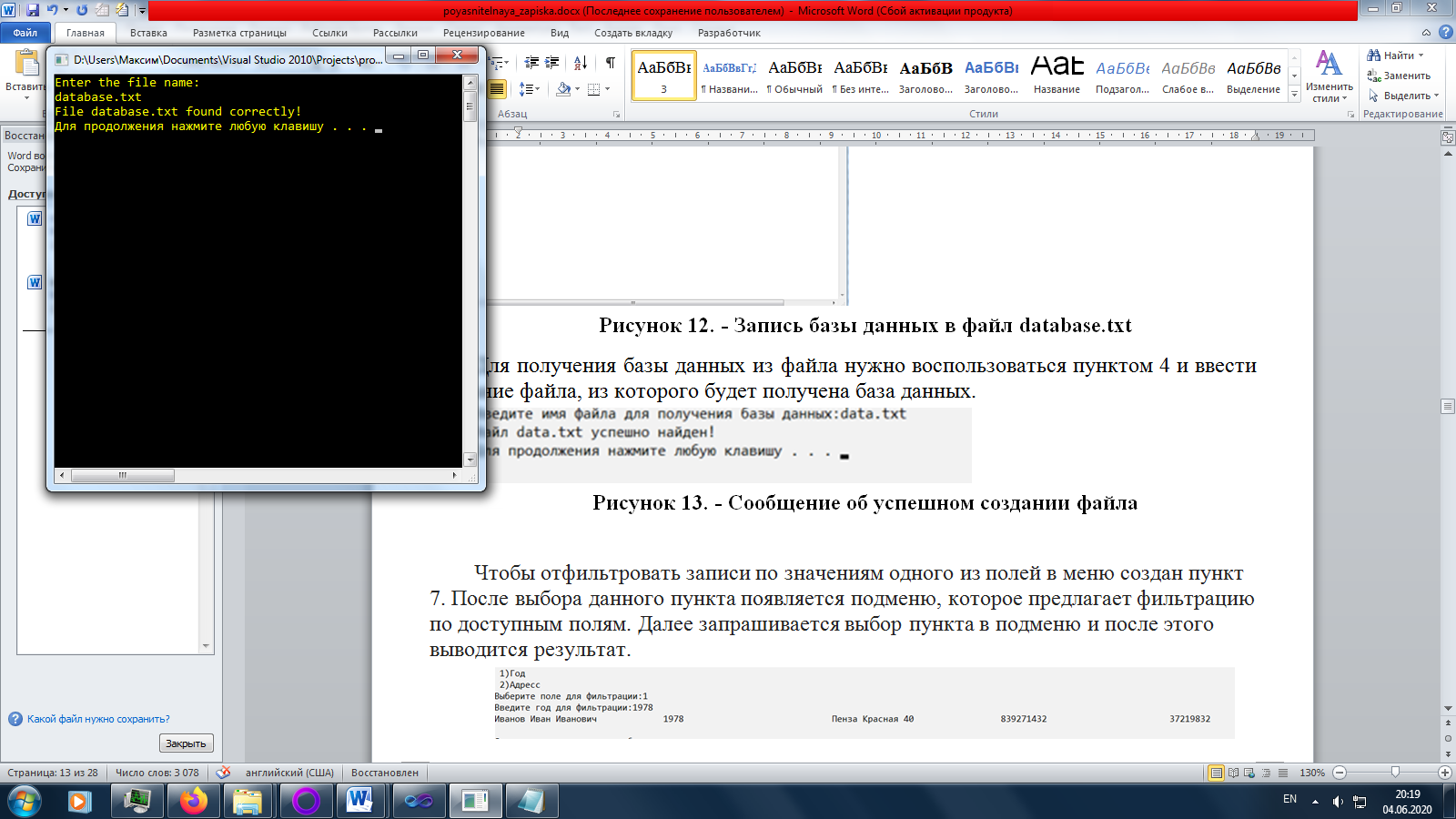


Рисунок 12. - Сообщение об успешном создании файла

Чтобы отфильтровать записи по значениям одного из полей в меню создан пункт 7. После выбора данного пункта появляется подменю, которое предлагает фильтрацию по доступным полям. Далее запрашивается выбор пункта в подменю и после этого выводится результат. Выбираем фильтрацию по жанру

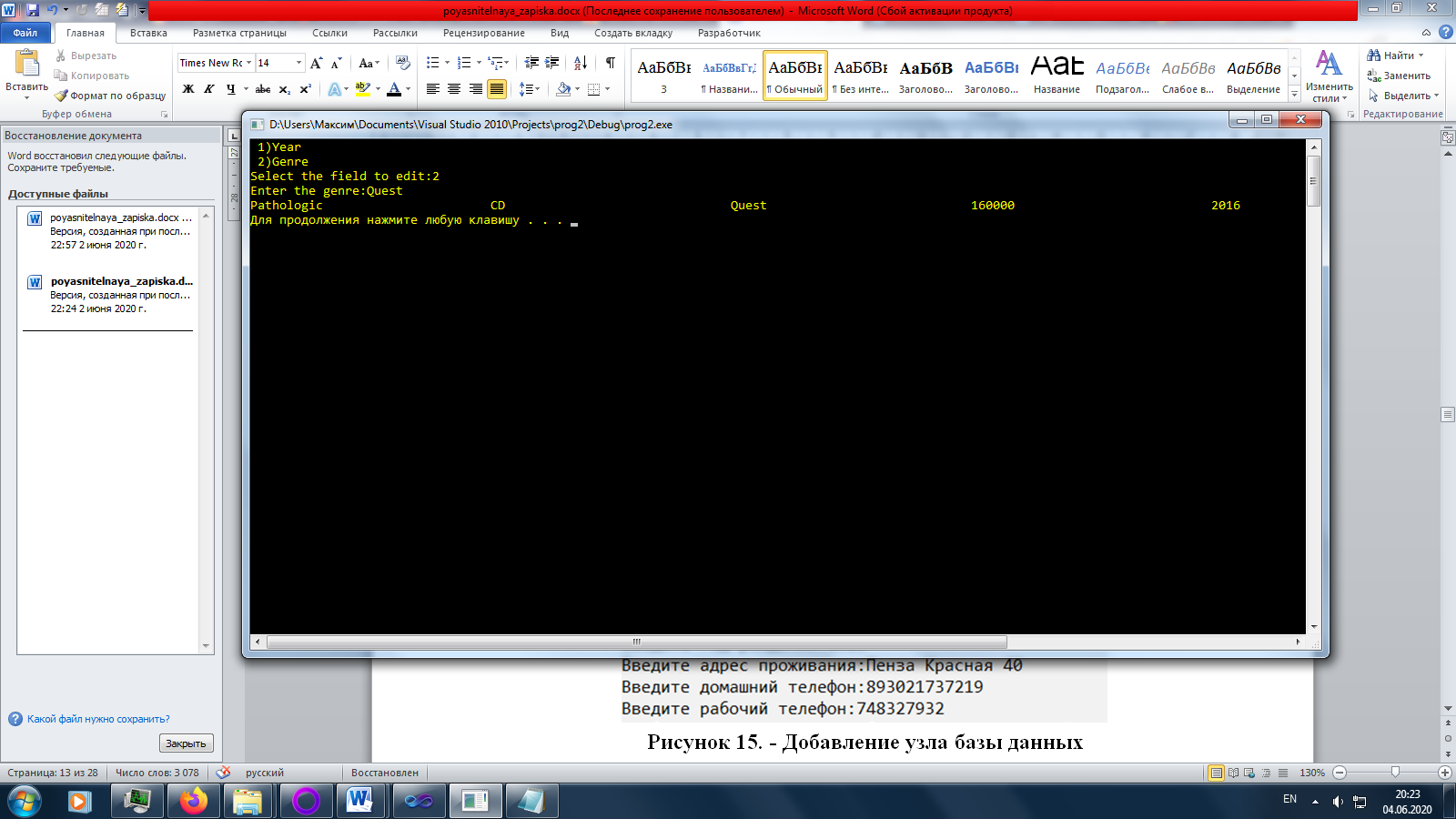
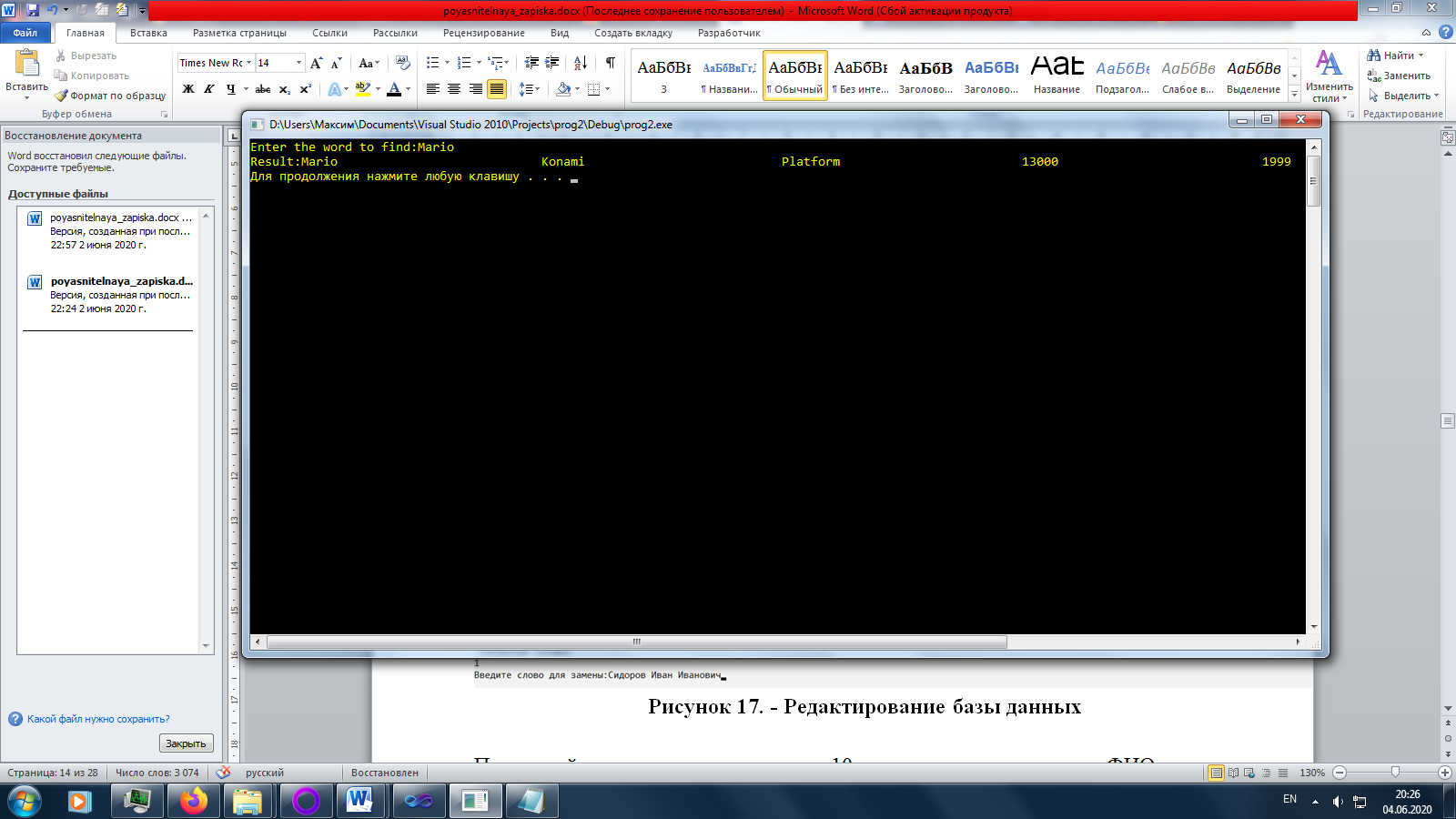


Рисунок 13. –Фильтрация

Для удаления последнего узла базы данных предназначен пункт меню под номером5.

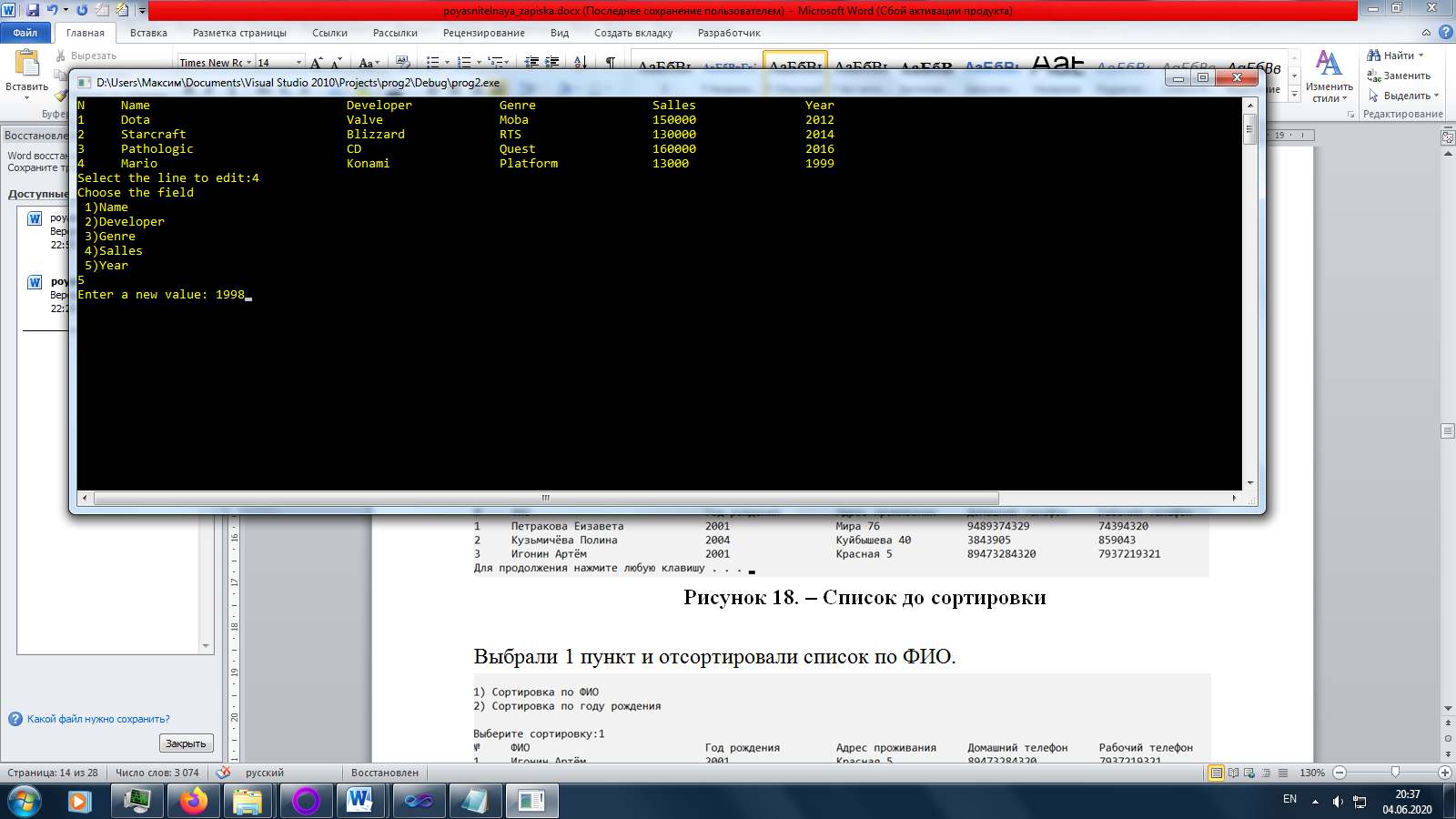
Если узел необходимо добавить, то следует воспользоваться 4-ым пунктом меню. Этот пункт позволяет добавлять узел в конец базы данных.

Пункт 8 в главное меню осуществляет поиск по слову.  
Пользователь должен ввести слово, после чего выведется результат.



**Рисунок 14. -Поиск по слову**

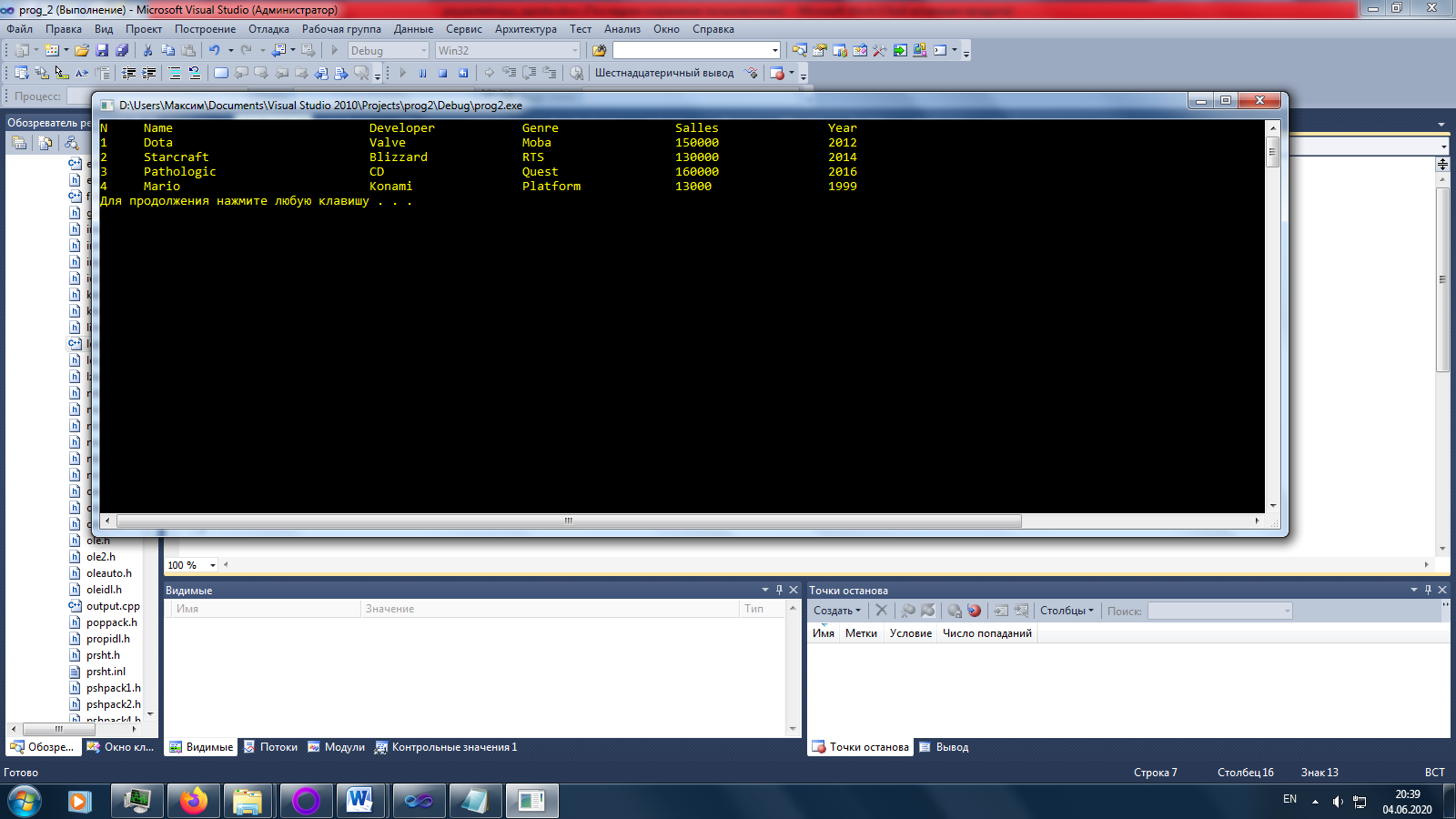
Пункт 10 в главном меню позволяет редактировать записи в базе данных.





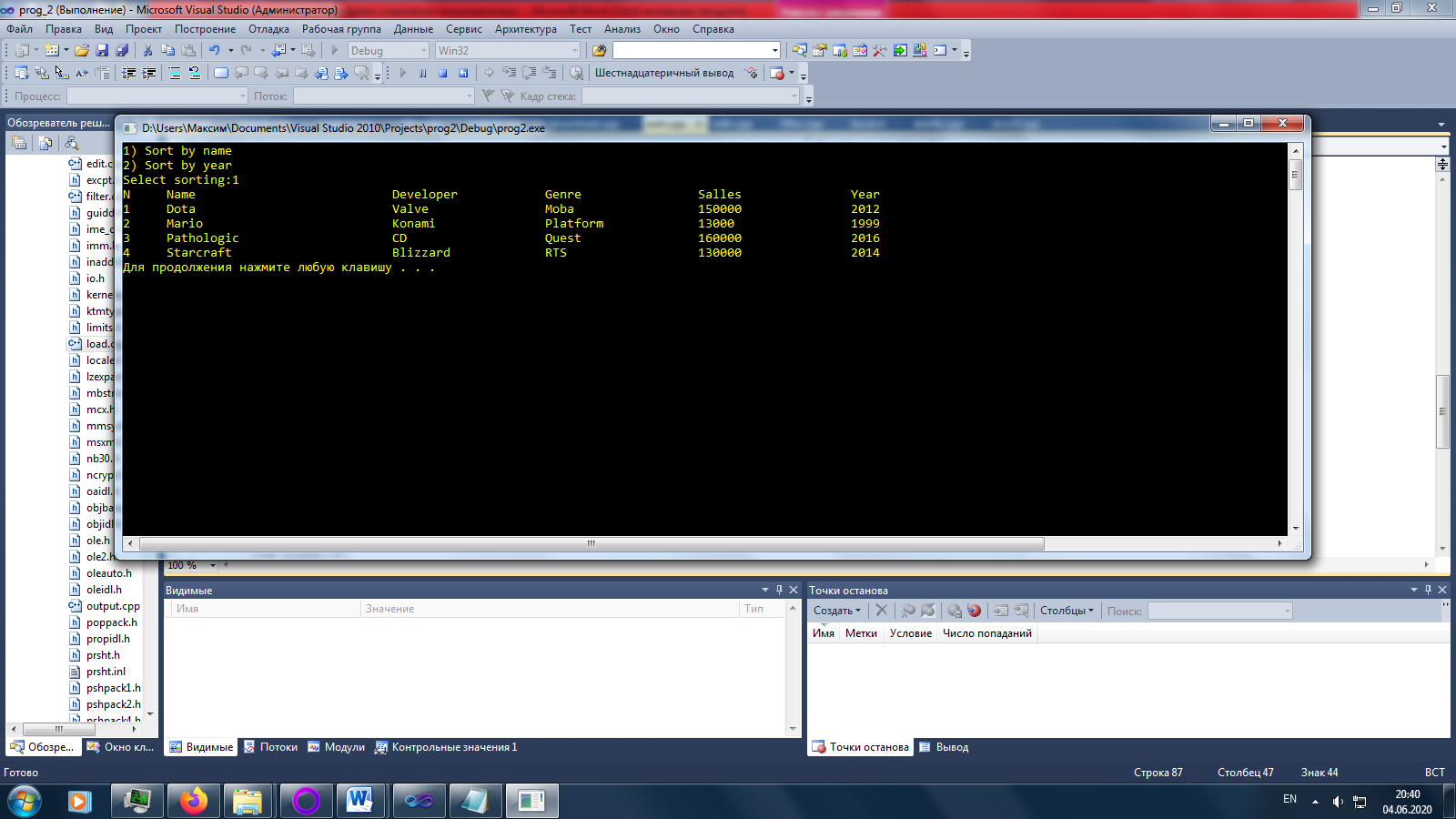
**Рисунок 15. -Редактирование базы данных (до и после)**

Пункт меню под номером 9 выполняет сортировку по Названию или по Году выпуска.



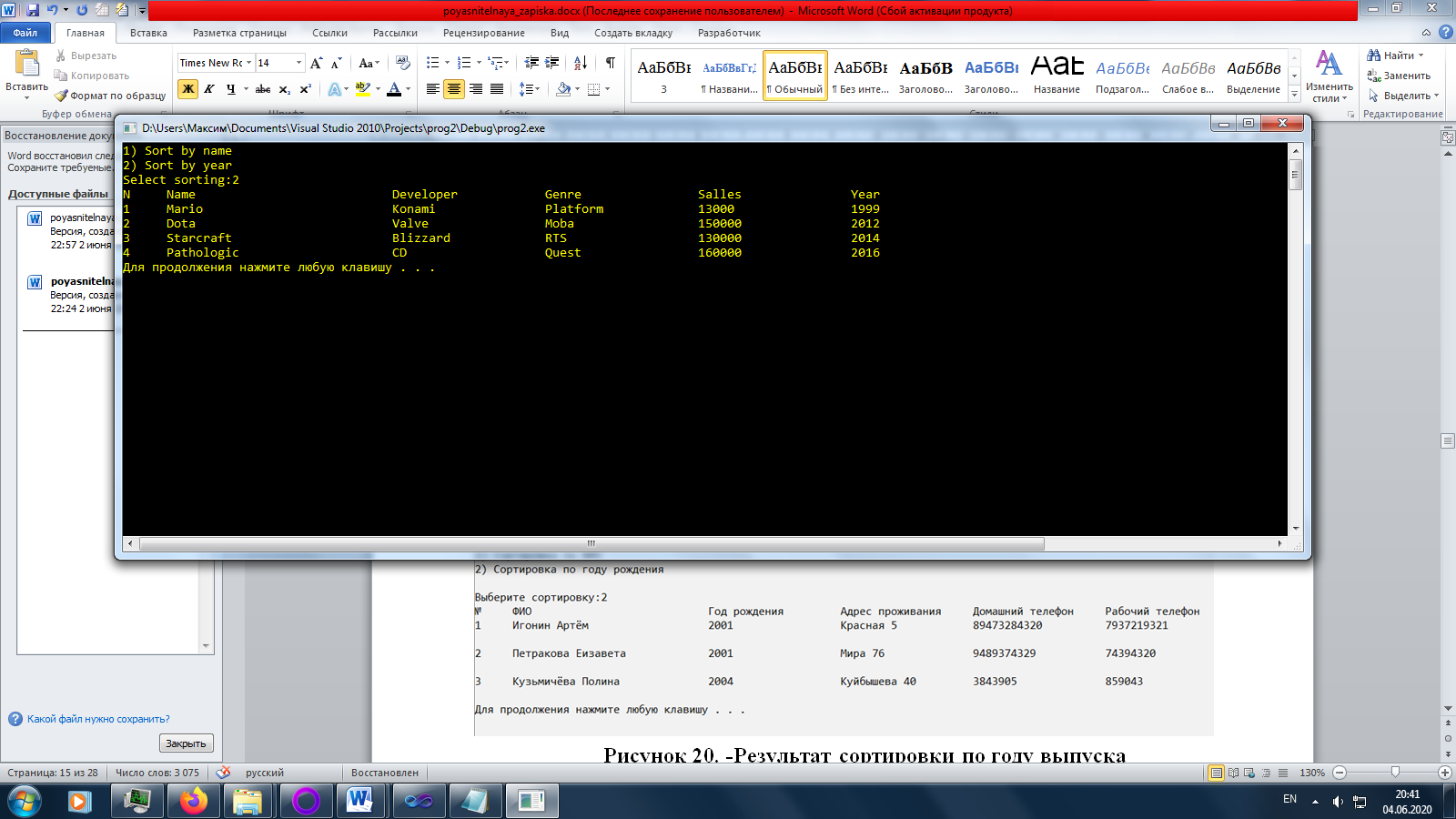
**Рисунок 16. –Список до сортировки**

Выбрали 1 пункт и отсортировали список по Названию.



**Рисунок 17. -Результат сортировки по Названию**

Выбрали 2 пункт и отсортировали по году выпуска.



**Рисунок 18. -Результат сортировки по году выпуска**

Последний пункт 11 осуществляет выход из программы.

# Заключение

При выполнении данной курсовой работы были получены навыки разработки многомодульных программ. Были освоены приемы создания псевдографического интерфейса, изучены функции работы с консолью, способы обработки событий от клавиатуры. Усвоены механизмы реализации меню. Изучены функции работы с файлами. Получены базовые навыки программирования на языке Си. Изучены основные возможности среды программирования VisualStudio2019. Получены навыки отладки и тестирования программ.

В рамках выполнения курсовой работы была написана программа, необходимая для ведения базы данныхкомпьютерных игр. Программа предоставляет небольшой, но достаточный для использования список возможностей. Не требует большого количества системных ресурсов и обеспечивает быструю обработку базы данных.

# Список используемых источников

1. http://cppstudio.com
2. https://cyberforum.ru
3. Герберт Шилдт. Полный справочник по С.
4. http://programmersforum.ru
5. http://ru.wikipedia.org

# Приложение

**Файл database.txt**

**Name Developer Genre Salles Year**

**Mario Konami Platform 13000 1999**

**Pathologic CD Quest 160000 2016**

**Starcraft Blizzard RTS 130000 2014**

**Dota Valve Moba 150000 2012**

# Листинг

Файлgames.h

#include<iostream>

#include<string.h>

#include<stdlib.h>

#include<stdio.h>

#include"struct.cpp"

void sort(game\_list\*\* head);

void sort\_nazvanie(game\_list\*\* head);

void sort\_year(game\_list\*\* head);

void spisok(game\_list\*\* head);

void download(game\_list\*\* head);

void title();

void del(game\_list\*\* head);

void search(game\_list\*\* head);

void filter(game\_list\*\* head);

void edit(game\_list\*\* head);

void load(game\_list\*\* head);

void download(game\_list\*\* head);

void pri(game\_list\*\* head);

Файл main\_f.cpp(с ассемблерной вставкой)

#define \_CRT\_SECURE\_NO\_WARNINGS

#include"games.h"

#include<stdio.h>

#include<conio.h>

#include<string.h>

#include<stdlib.h>

#include<locale.h>

#include<io.h>

#include<Windows.h>

#include"games.h"

void main()

{

SetConsoleCP(1251);

SetConsoleOutputCP(1251);

title();

system("pause");

int count;

game\_list\* head = NULL;

for (;;)

{

system("cls");

char menu[]=(" ----------------------------------\n | 1.Create date base |\n ----------------------------------\n | 2.Load | \n ----------------------------------\n | 3.Get from file |\n ----------------------------------\n | 4.Add |\n ----------------------------------\n | 5.Delete the last line |\n ----------------------------------\n | 6.Data base output |\n ----------------------------------\n | 7.Filter |\n ----------------------------------\n | 8.Search |\n ----------------------------------\n | 9.Sort |\n ----------------------------------\n | 10.Edit |\n ----------------------------------\n | 11.Exit | \n ----------------------------------\n \n");

\_\_asm

{

xor eax, eax

lea eax, menu

push eax

call dword ptr[printf]

add esp, 4

}

printf(" \n Enter the operation number:");

scanf("%d", &count);

switch (count)

{

case 1:

system("cls");

spisok(&head);

break;

case 2:

system("cls");

load(&head);

break;

case 3:

system("cls");

download(&head);

system("pause");

break;

case 4:{

system("cls");

spisok(&head);

system("pause");

break;

}

case 5:

system("cls");

del(&head);

system("pause");

break;

case 6:

system("cls");

fflush(stdin);

pri(&head);

system("pause");

break;

case 7:

system("cls");

filter(&head);

system("pause");

break;

case 8:

system("cls");

search(&head);

system("pause");

break;

case 9:

system("cls");

sort(&head);

system("pause");

break;

case 10:

system("cls");

edit(&head);

pri(&head);

break;

case 11:

exit(0);

break;

}

}

}

Файлdownload.cpp

#include"games.h"

void download(game\_list\*\* head)

{

int i;

FILE \* S;

char name[32], nazvanie[32], developer[32], genre[32], sells[32], year[32];

int fd;

\*head = NULL;

fprintf(stdout, "%s", "Enter the file name:\n");

scanf("%s", &name);

while ((S = fopen(name, "r+")) == (NULL))

{

fprintf(stdout, "%s", "File not found!\nTry again:");

scanf("%s", &name);

}

fprintf(stdout, "File %s found correctly!\n", name);

if (!feof(S))

{

fseek(S, 100, SEEK\_SET);

while (!feof(S))

{

fscanf(S,"%s",&nazvanie);

fscanf(S,"%s",&developer);

fscanf(S,"%s",&genre);

fscanf(S,"%s",&sells);

fscanf(S,"%s",&year);

if(!feof(S))

push(head, nazvanie, developer, genre, sells, year);

}

}

else

{

printf("\nEMPTY!\n\n");

}

fclose(S);

}

Файлload.cpp

#include"games.h"

void load(game\_list\*\* head)

{

FILE \*S;

game\_list\* ptr = (\*head);

char name[32];

fflush(stdin);

fprintf(stdout, "%s", "Emter the name of file:");

scanf("%s", &name);

while ((S = fopen(name, "w")) == (NULL))

{

fprintf(stdout, "%s", "File not found!\n");

scanf("%s", &name);

}

fprintf(stdout, "The file %s was found successfully!\n", name);

if ((\*head) != NULL)

{

game\_list \*ptr = \*head;

fprintf(S, "\n\n\n%-25s %-20s %-20s %-20s %-20s\n", "Name", "Developer", "Genre", "Salles", "Year");

while (ptr != NULL)

{

fprintf(S, "%-25s %-20s %-20s %-20s %-20s\n", ptr->nazvanie, ptr->developer, ptr->genre, ptr->sells, ptr->year);

ptr = ptr->next;

}

}

else

{

printf("\EMPTY\n\n");

}

fclose(S);

}

Файлsearch.cpp

#include"games.h"

void search(game\_list\*\* head)

{

char name[16];

game\_list\* ptr = (\*head);

fflush(stdin);

printf("Enter the word to find: ");

gets(name);

if (ptr == NULL)

printf("EMPTY");

else

printf("Result:");

while (ptr != NULL)

{

if (strstr(ptr->nazvanie, name) != NULL)

printf("%-32s %-32s %-32s %-32s %-32s\n", ptr->nazvanie, ptr->developer, ptr->genre, ptr->sells, ptr->year);

if (strstr(ptr->developer, name) != NULL)

printf("%-32s %-32s %-32s %-32s %-32s\n", ptr->nazvanie, ptr->developer, ptr->genre, ptr->sells, ptr->year);

if (strstr(ptr->genre, name) != NULL)

printf("%-32s %-32s %-32s %-32s %-32s\n", ptr->nazvanie, ptr->developer, ptr->genre, ptr->sells, ptr->year);

if (strstr(ptr->sells, name) != NULL)

printf("%-32s %-32s %-32s %-32s %-32s\n", ptr->nazvanie, ptr->developer, ptr->genre, ptr->sells, ptr->year);

if (strstr(ptr->year, name) != NULL)

printf("%-32s %-32s %-32s %-32s %-32s\n", ptr->nazvanie, ptr->developer, ptr->genre, ptr->sells, ptr->year);

ptr = ptr->next;

}

}

Файлtitle.cpp

#include"games.h"

void title()

{

printf ( "Курсовая работа по дисциплине Программирование\nБаза данных Список компьютерных игр\nВыполнил студент группы 21ВВ4:\n\Калыева С.К.\n\n");

\_getch();

}

Файлoutput.cpp

#include"games.h"

void pri(game\_list\*\* head)

{

if (\*head != NULL)

{

int i = 1;

game\_list\* ptr = (\*head);

ptr = (\*head);

printf("N %-30s %-20s %-20s %-20s %-20s\n", "Name", "Developer", "Genre", "Salles", "Year");

while (ptr != NULL)

{

printf("%-5d %-30s %-20s %-20s %-20s %-20s\n", i, ptr->nazvanie, ptr->developer, ptr->genre, ptr->sells, ptr->year);

ptr = ptr->next;

i++;

}

}

else

{

printf("EMPTY");

}

}

Файлsort.cpp

#include"games.h"

#define \_CRT\_SECURE\_NO\_WANINGS

void sort\_year(game\_list\*\* head)

{

char nazvanie[16], developer[16], genre[16], sells[16], year[16];

int size = 0;

game\_list\* ptr = (\*head);

while (ptr != NULL)

{

ptr = ptr->next;

size++;

}

ptr = (\*head);

for (int j = 1; j < size; j++)

{

ptr = (\*head);

for (int i = 0; i < size - j; i++)

{

if (atoi(ptr->year) > atoi(ptr->next->year))

{

strcpy(nazvanie, ptr->nazvanie);

strcpy(developer, ptr->developer);

strcpy(genre, ptr->genre);

strcpy(sells, ptr->sells);

strcpy(year, ptr->year);

strcpy(ptr->nazvanie, ptr->next->nazvanie);

strcpy(ptr->developer, ptr->next->developer);

strcpy(ptr->genre, ptr->next->genre);

strcpy(ptr->sells, ptr->next->sells);

strcpy(ptr->year, ptr->next->year);

strcpy(ptr->next->nazvanie, nazvanie);

strcpy(ptr->next->developer, developer);

strcpy(ptr->next->genre, genre);

strcpy(ptr->next->sells, sells);

strcpy(ptr->next->year, year);

}

ptr = ptr->next;

}

}

}

void sort\_nazvanie(game\_list\*\* head)

{

char nazvanie[16], developer[16], genre[16], sells[16], year[16];

int size = 0;

game\_list\* ptr = (\*head);

while (ptr != NULL)

{

ptr = ptr->next;

size++;

}

ptr = (\*head);

for (int j = 1; j < size; j++)

{

ptr = (\*head);

for (int i = 0; i < size - j; i++)

{

if (ptr->nazvanie[0] > ptr->next->nazvanie[0])

{

strcpy(nazvanie, ptr->nazvanie);

strcpy(developer, ptr->developer);

strcpy(genre, ptr->genre);

strcpy(sells, ptr->sells);

strcpy(year, ptr->year);

strcpy(ptr->nazvanie, ptr->next->nazvanie);

strcpy(ptr->developer, ptr->next->developer);

strcpy(ptr->genre, ptr->next->genre);

strcpy(ptr->sells, ptr->next->sells);

strcpy(ptr->year, ptr->next->year);

strcpy(ptr->next->nazvanie, nazvanie);

strcpy(ptr->next->developer, developer);

strcpy(ptr->next->genre, genre);

strcpy(ptr->next->sells, sells);

strcpy(ptr->next->year, year);

}

ptr = ptr->next;

}

}

}

void sort(game\_list\*\* head)

{

int a;

printf("1) Sort by name\n2) Sort by year\nSelect sorting:");

scanf("%d", &a);

switch (a)

{

case 1:

sort\_nazvanie(head);

pri(head);

system("pause");

break;

case 2:

sort\_year(head);

pri(head);

system("pause");

break;

default:

break;

}

}

Файлedit.cpp

#include"games.h"

void edit(game\_list\*\* head)

{

game\_list\* ptr = (\*head);

int i = 0;

char name[16];

pri(head);

if (ptr != NULL)

{

ptr = (\*head);

printf("Select the line to edit:");

scanf("%d", &i);

i--;

for (; i > 0; i--)

{

ptr = ptr->next;

if (ptr == NULL)

{

exit(-1);

printf("Error when entering!");

}

}

printf("Choose the field\n 1)Name \n 2)Developer \n 3)Genre \n 4)Salles \n 5)Year \n");

scanf("%d", &i);

fflush(stdin);

printf("Enter a new value: ");

gets(name);

switch (i)

{

case 1:

strcpy(ptr->nazvanie, name);

break;

case 2:

strcpy(ptr->developer, name);

break;

case 3:

strcpy(ptr->genre, name);

break;

case 4:

strcpy(ptr->sells, name);

break;

case 5:

strcpy(ptr->year, name);

break;

default: break;

}

}

else {

printf("EMPTY!");

}

}

Файлfilter.cpp

#include"games.h"

void filter(game\_list\*\* head)

{

game\_list\* ptr = (\*head);

fflush(stdin);

fflush(stdout);

char number[16];

int i;

printf(" 1)Year\n 2)Genre\nSelect the field to edit:");

scanf("%d", &i);

fflush(stdin);

switch (i)

{

case 1:

fflush(stdin);

printf("Enter the year:");

gets(number);

while (ptr != NULL)

{

if (strstr(ptr->year, number) != NULL)

printf("%-32s %-32s %-32s %-32s %-32s\n", ptr->nazvanie, ptr->developer, ptr->genre, ptr->sells, ptr->year);

ptr = ptr->next;

}

break;

case 2:

fflush(stdin);

printf("Enter the genre:");

gets(number);

while (ptr != NULL)

{

if (strstr(ptr->genre, number) != NULL)

printf("%-32s %-32s %-32s %-32s %-32s\n", ptr->nazvanie, ptr->developer, ptr->genre, ptr->sells, ptr->year);

ptr = ptr->next;

}

break;

default: printf("Wrong input");

}

}

Файлdel.cpp

#include"games.h"

void del(game\_list\*\* head)

{

if ((\*head) != NULL)

{

game\_list\* prev = NULL;

prev = (\*head);

(\*head) = (\*head)->next;

free(prev);

}

else

{

printf("\nThe list is empty!\n");

}

}

Файлstruct.cpp

#include"games.h"

typedefstruct game\_list

{

char nazvanie[16],

developer[16],

genre[16],

sells[16],

year[16];

game\_list\* next;

};

Файлspis.cpp

#include"games.h"

void spisok(game\_list\*\* head)

{

fflush(stdin);

char nazvanie[32], developer[32], genre[32], sells[32], year[32];

printf("Enter the name:");

gets(nazvanie);

for (;;)

{

fflush(stdin);

printf("Enter the developer:");

gets(developer);

printf("Enter the genre:");

gets(genre);

printf("Enter sales:");

gets(sells);

printf("Enter the year:");

gets(year);

push(head, nazvanie, developer, genre, sells, year);

printf("To finish enter: `\nEnter the name:");

gets(nazvanie);

if (nazvanie[0] == '`')

break;

}

}

Файлadd.cpp

#include"games.h"

void push(game\_list\*\* head, char\* nazvanie, char\* developer, char\* genre, char\* sells, char\* year)

{

game\_list\* tmp;

tmp = new game\_list;

strcpy\_s(tmp->nazvanie, nazvanie);

strcpy\_s(tmp->developer, developer);

strcpy\_s(tmp->genre, genre);

strcpy\_s(tmp->sells, sells);

strcpy\_s(tmp->year, year);

tmp->next = (\*head);

(\*head) = tmp;

}