МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ РОССИЙСКОЙ ФЕДЕРАЦИИ

федеральное государственное автономное образовательное учреждение высшего образования

«САНКТ-ПЕТЕРБУРГСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ   
АЭРОКОСМИЧЕСКОГО ПРИБОРОСТРОЕНИЯ»

КАФЕДРА №14

КУРСОВАЯ РАБОТА  
ЗАЩИЩЕНА С ОЦЕНКОЙ

РУКОВОДИТЕЛЬ

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| доц., канд. техн. наук |  |  |  | Курицын К.А. |
| должность, уч. степень, звание |  | подпись, дата |  | инициалы, фамилия |

|  |
| --- |
| ПОЯСНИТЕЛЬНАЯ ЗАПИСКА К КУРСОВОЙ РАБОТЕ |
| ПАТТЕРНОЕ ПРОЕКТИРОВАНИЕ |
| по дисциплине: ТЕХНОЛОГИЯ ПРОГРАММИРОВАНИЯ |
|  |
|  |

РАБОТУ ВЫПОЛНИЛ

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| СТУДЕНТ ГР. № | 1641 |  |  |  | П.О. Винарь |
|  |  |  | подпись, дата |  | инициалы, фамилия |

Санкт-Петербург 2019

Содержание

1. Задание …………………………………………………………………………...3
2. Функциональные требования…………………………………………………...3
3. Технические спецификации……………………………………………………..3
4. Взаимодействие с внешними и внутренними системами……………………..5
5. Программа и методика испытаний……………………………………………...5
6. Литература………………………………………………………………………..6

Приложение 1 – Описание формата………………………………………………...7

Приложение 2 – Листинг………………………………………………………….....7

1. Листинг Game.h ……………………………………………………………...8
2. Листинг Game.cpp ……………………………………………………...……8
3. Листинг Chain.h………………………………………………………………9
4. Листинг Chain. Cpp ………………………………………………………….10
5. Листинг Shop.h ………………………………………………………………13
6. Листинг Shop.cpp ……………………………………………………………13
7. Листинг Site.h ………………………………………………………………..13
8. Листинг Site.cpp ………………………………………………………...…...14
9. Листинг Warehose.h ……………………………………………………...….14
10. Листинг Warehose.cpp …………………………………………………...….14
11. Листинг main.cpp …………………………………………………………….15
12. Листинг Header.h …………………………………………………………….17

Приложение 3 – Примеры, скриншоты……………………………………………..18

1. **Задание**

Создать систему запросов «Сайт → Магазин → Склад». Товар, который интересует клиента, – компьютерные игры, которые имеют ряд характеристик: название, жанр, год выпуска, платформа (ПК, Xbox, PS), требования к платформе, разработчик, количество. Пользователь использует сайт для нахождения интересующего товара, затем запрос отправляется в магазин, затем на склад. Использовать паттерн «Цепочка обязанностей». Данные о товаре для сайта, магазина и склада берутся из внешних входных файлов «site», «shop» и «warehouse» соответственно.

1. **Функциональные требования**

2.1) Выход из приложения

Реализовать выход из приложения.

2.2) Просмотр каталогов игр.

2.3) Добавление компьютерных игр.

2.3.1) Добавление на сайт. Заполнение файла через консоль.

2.3.2) Добавление в магазин. Заполнение файла через консоль.

2.3.3) Добавление на склад. Заполнение файла через консоль.

Реализовать добавление компьютерных игр в файл-каталог. Добавление осуществляется через консоль

2.4) Поиск компьютерных игр.

Реализовать поиск указанной компьютерной игры из каталога сайта/магазина/склада по названию, игровой платформе и количеству (должно быть меньше чем в текущем каталоге).

2.5) Сохранение изменений в файл.

Результат записывается в txt файл формата указанного в Приложении 1.

2.6) Пользовательское меню.

Интерфейс должен быть удобным и понятным.

2.7) Язык С++

Приложение должно быть реализовано на языке С++. Должно быть разделение на .cpp и .h файлы.

2.8) Необходимо использовать паттерн «Цепочка обязанностей»

1. **Технические спецификации**

3.1) При запуске приложения перед пользователем отображается название приложения и варианты действий для работы.

* |1|<->Вывести все данные из всех источников
* |2|<->Добавление элементов в базу
* |3|<->Найти игру на сайте->магазине->складе
* |0|<->Выход

3.2) Нулевой пункт отвечает за выход из приложения.

При выборе пункта 0, осуществляется выход из приложения.

3.3) Первый пункт отвечает вывод всех каталогов.

Если выбран первый пункт, то все данные из файлов (site, shop,wirehouse), выводятся на экран. При этом после вывода на экран, программа возвращается в главное меню.

3.4) Второй пункт отвечает за добавление элементов в базу.

При выборе пункта 2, и нажатия клавиши Enter, осуществляется еще один запрос, в какой каталог будет происходить добавление (сайт/магазин /склад). Выводится следующее подменю:

"|1| <-> Добавление игр на сайт" << endl <<

"|2| <-> Добавление игр в магазин" << endl <<

"|3| <-> Добавление игр на склад" << endl <<

"|0| <-> Отмена" << endl;

3.4.1) Добавление элементов на сайт.

При выборе из подменю пункта 1, на экран выведется содержимое из файла site. Далее происходит запрос сколько игр необходимо добавить на сайт. Вводится количество, и далее все характеристики, после каждого запроса. После ввода всех данных, новое содержимое файла site выводится на экран. После чего, пользователь вновь попадает в основное меню.

3.4.2) Добавление элементов в магазин.

При выборе из подменю пункта 2, на экран выведется содержимое из файла shop. Далее происходит запрос сколько игр необходимо добавить в магазин. Вводится количество, и далее все характеристики, после каждого запроса. После ввода всех данных, новое содержимое файла shop выводится на экран. После чего, пользователь вновь попадает в основное меню.

3.4.3) Добавление элементов на склад.

При выборе из подменю пункта 3, на экран выведется содержимое из файла wirehouse. Далее происходит запрос сколько игр необходимо добавить на склад. Вводится количество, и далее все характеристики, после каждого запроса. После ввода всех данных, новое содержимое файла wirehouse выводится на экран. После чего, пользователь вновь попадает в основное меню.

3.4.4) Выход из подменю.

При выборе пункта ноль, произойдет переход в основное меню.

3.5) Третий пункт отвечает за поиск компьютерной игры.

При выборе третьего пункта, происходит поиск игры, по трем основным критериям:

* Название
* Платформа
* Количество

При нажатии клавиши Enter выводится на экран сообщение, где была найдена игра. После чего пользователь вновь попадает в основное меню.

3.6) Описание паттерна «Цепочка обязанностей».

Класс Game, служит для хранения всех игр. Это независимый класс.

Базовый класс Chain, это класс обработчиков, в котором содержится метод получения следующего обработчика. В базовом классе существует обработчик, поле для хранения ссылки на следующий объект цепочки. Программа предполагает динамическую перестройку цепочек, существует сеттер для поля.

Базовый метод обработки перенаправляет запрос следующему объекту, для поиска требуемой игры, что дает возможность передавать запросы дальше по цепи, обращаясь к родительской реализации метода.

Также существует классы наследники, в которых будет осуществляться поиск:

Site, Shop, Wirehose.

Запрос посылается от первого обработчика, а далее идет поиск, где находится компьютерная игра.

Запросы могут достигать конца, оставаясь необработанными.

1. **Взаимодействия с внешними и внутренними системами**

Взаимодействие типа консоль-файл. Приложение взаимодействует с тремя файлами:

Site, Shop, Warehouse.

1. **Программа и методика испытаний**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| № | Сценарий проверки | Ожидаемый результат | ФТ | Тех. Спец. | Результат |
| 1 | Включение приложения | Отображение названия программы и всех пунктов для работы с файлом каталога:  |1|<->Вывести все данные из всех источников  |2|<->Добавление элементов в базу  |3|<->Найти игру на сайте->магазине->складе  |0|<->Выход | - | 3.1 | Готово  Рисунок 1 |
| 2 | В меню выбрать 0 | Выход из приложения | 2.1 | 3.2 | Готово  Рисунок 2 |
| 3 | В меню выбрать 1 | Просмотр каталогов игр | 2.2 | 3.3 | Готово  Рисунок 3 |
|  |  | После выведенного результата возврат в главное меню | - | 3.3 | Готово |
| 4 | В меню выбрать 2 | Добавление компьютерных игр | 2.3 | 3.4 | Готово  Рисунок 4 |
|  | В подменю выбрать 1 | Добавление на сайт | 2.3.1 | 3.4.1 | Готово  Рисунок 5 |
|  |  | После выведенного результата возврат в главное меню | - | 3.3 | Готово |
|  | В подменю выбрать 2 | Добавление в магазин | 2.3.2 | 3.4.2 | Готово  Рисунок 6 |
|  |  | После выведенного результата возврат в главное меню | - | 3.3 | Готово |
|  | В подменю выбрать 3 | Добавление на склад | 2.3.3 | 3.4.3 | Готово  Рисунок 7 |
|  | В подменю выбрать 0 | Отмена | - | 3.4.4 | Готово |
| 5 | В меню выбрать 3 | Запрос на поиск | 2.4 | 3.5 | Готово  Рисунок 8 |
|  |  | Результат(если такой товар только на сайте) | - | 3.5 | Готово  Рисунок 9 |
|  |  | Результат(если такой товар только в магазине) | - | 3.5 | Готово  Рисунок 10 |
|  |  | Результат (если такой товар только на складе) | - | 3.5 | Готово  Рисунок 11 |
|  |  | Результат(если товар есть и на сайте и в магазине) | - | 3.5 | Готово  Рисунок 12 |
|  |  | Результат( если товар есть и на сайте и на складе) | - | 3.5 | Готово  Рисунок 13 |
|  |  | Результат (если товар есть везде) | - | 3.5 | Готово  Рисунок 14 |
|  |  | Результат (если товар есть в магазине и на складе) | - | 3.5 | Готово  Рисунок 15 |
|  |  | Результат (если товар есть на сайте и в магазине, но количество не удовлетворяет) | - | 3.5 | Готово  Рисунок 16 |
|  |  | Результат( если товар есть на сайте и на складе, но количество не удовлетворяет) | - | 3.5 | Готово  Рисунок 17 |
|  |  | Результат(товара нет нигде) |  | 3.5 | Готово  Рисунок 18 |

1. **Литература**

* <https://ru.wikipedia.org/wiki/Цепочка_обязанностей#Пример_на C++>
* http://www.cyberforum.ru
* https://ru.stackoverflow.com
* <http://cppstudio.com>
* <http://www.sugardas.lt/~p2d/books/Priemioop.pdf>
* http://www.c-cpp.ru
* Шилдт Г. Самоучитель С++
* Бьерн Страуструп - Язык программирования С++

**Приложение №1 Описание формата**

Формат файла должен отвечать следующим требованиям:

1. На первом месте в файле находится целое число, означающее количество игр в файле. Все файлы заполняются в одинаковом порядке.
2. Далее по порядку расположены:

* Название игры(массив char из не более 50 символов). Название игры вводится только латиницей, а также не допускаются пробелы между словами. Для разделения требуется использовать «\_»;
* Жанр (массив char из не более 20 символов). Требуется вводить только латиницей, а также не допускаются пробелы;
* Год выпуска (целое число int). Вводится только год. Например, 2012.
* Игровая платформа (массив char состоящий из не более 10 символов). Требуется вводить только латиницей, а также не допускаются пробелы;
* Разработчик (массив char состоящий из не более 30 символов). Требуется вводить только латиницей, а также не допускаются пробелы;
* Количество данной игры(целое число).

1. Информация об игре указывается столько раз, сколько игр должно быть в файле.
2. Текстовый файл имеет кодировку UTF-8.
3. Информация для каждого количества записывается в строку. Если требуется записать снова данные, то сначала переход на новую строку, затем вводится количество, далее все данные вводятся по порядку.

**Приложение №2 Листинг программы.**

Ссылка на проект в <https://github.com/polinavinar/kurs/tree/master/Kurs_Cep>

1. **Листинг Game.h**

#pragma once

class Game {

friend class Сhain;

char name[50]; //название

char genre[20]; //жанр

int year; //год выпуска

char platform[10]; //игровая платформа (ПК, Xbox, PS)

char sys\_req[100]; //системные треб

char developer[30]; //разработчик

int quantity; //количество

public:

///конструктор

Game();

///конструктор с параметрами

Game(char\* name1, char\* genre1, int year1, char\* platform1, char\* sys\_req1, char\* developer1, int quantity1);

///Сеттер

void set\_game(void);

///просмотр

void show\_game(void);

};

1. **Листинг Game.cpp**

#include "Header.h"

#include "Game.h"

using namespace std;

Game::Game()

{

name[0] = EOF;

genre[0] = EOF;

year = 0;

platform[0] = EOF;

sys\_req[0] = EOF;

developer[0] = EOF;

quantity = 0;

}

Game::Game(char \* name1, char \* genre1, int year1, char \* platform1, char \* sys\_req1, char \* developer1, int quantity1)

{

strcpy(name, name1);

strcpy(name, name1);

year = year1;

strcpy(name, name1);

strcpy(name, name1);

strcpy(name, name1);

quantity = quantity1;

}

void Game::set\_game(void)

{

cout << "Введите название " << endl;

cin >> name;

cout << "Введите жанр " << endl;

cin >> genre;

cout << "Введите год выпуска " << endl;

cin >> year;

cout << "Введите игровую платформу " << endl;

cin >> platform;

cout << "Введите системные требования " << endl;

cin >> sys\_req;

cout << "Введите разработчика " << endl;

cin >> developer;

cout << "Введите количество " << endl;

cin >> quantity;

}

void Game::show\_game(void)

{

cout << "Название игры: " << name << "; Жанр игры: " << genre << "; Год выпуска " << year << "; Игровая платформа " << platform << "; Cистемные требования " << sys\_req << endl;

cout << "; Разработчик " << developer << "; Количество " << quantity << ";" << endl;

}

1. **Листинг Chain.h**

#pragma once

#include "Header.h"

#include "Game.h"

class Сhain {

protected:

Сhain \* next;

Game \*game\_list; // массив с играми

int size\_of\_game\_list; // размер массива с играми

virtual void result(Game\* game); // результат найденной игры

public:

Сhain();

///Если не обнаружили нужную нам игру в этом узле, переходим на следующий

Сhain\* setNext(Сhain\* сhain);

/// Если не обнаружили нужную нам игру в этом узле, переходим на следующий

void search(Game\* game);

///дозапись в файл

void add\_to\_file(string file);

///запись в файл

void write\_to\_file(string file, int add\_val);

///чтение из файла

void read\_from\_file(string file);

virtual ~Сhain();

};

1. **Листинг Chain. Cpp**

#include "Header.h"

#include "Chain.h"

#include "Game.h"

void Сhain::result(Game \* game) {}

Сhain::Сhain()

{

next = nullptr;

game\_list = nullptr;

size\_of\_game\_list = NULL;

}

Сhain \* Сhain::setNext(Сhain \* сhain)

{

next = сhain;

return next;

}

void Сhain::search(Game \* game)

{

bool flag = false;

for (int i = 0; i < size\_of\_game\_list; i++)

{

if (!strcmp(game\_list[i].name, game->name) && !strcmp(game\_list[i].platform, game->platform) && (game\_list[i].quantity >= game->quantity))

{

flag = true;

}

}

if (flag)

{

result(game);

return;

}

else

{

if (next)

{

next->search(game);

}

else

{

cout << "Игра нигде не была найдена." << endl;

return;

}

}

}

void Сhain::add\_to\_file(string file)

{

int add\_value; //сколько игр хотим добавить

read\_from\_file(file);

cout << "Сколько игр вы хотите добавить в базу? " << endl;

cin >> add\_value;

size\_of\_game\_list += add\_value;

write\_to\_file(file, add\_value);

}

void Сhain::write\_to\_file(string file, int add\_val)

{

ofstream out;

Game tmp;

try

{

out.open(file, ios::out);

if (!out.is\_open()) {

throw 505;

}

out.write(reinterpret\_cast<char\*>(&size\_of\_game\_list), sizeof(int));

for (int i = 0; i < size\_of\_game\_list - add\_val; i++)

{

out.write(reinterpret\_cast<char\*>(&game\_list[i]), sizeof(Game)); //перезаписываем содержимое gamelist

}

for (int i = 0; i < add\_val; i++)

{

tmp.set\_game();

out.write(reinterpret\_cast<char\*>(&tmp), sizeof(Game)); //добавляем новые данные

}

out.close();

}

catch (const int &ex)

{

cout << "Ошибка открытия файла!" << endl << "Код ошибки: #" << ex << endl;

}

}

void Сhain::read\_from\_file(string file)

{

ifstream in;

try

{

in.open(file, ios::in);

if (!in.is\_open())

{

throw 505;

}

else

{

if (in.peek() == EOF)

{

throw 510;

}

}

in.read(reinterpret\_cast<char\*>(&size\_of\_game\_list), sizeof(int));

game\_list = new Game[size\_of\_game\_list];

for (int i = 0; i < size\_of\_game\_list; i++)

{

in.read(reinterpret\_cast<char\*>(&game\_list[i]), sizeof(Game));

game\_list[i].show\_game();

}

in.close();

}

catch (const int &ex)

{

cout << "Ошибка работы файла!" << endl << "Код ошибки: #" << ex << endl;

}

}

Сhain::~Сhain()

{

if (game\_list)

{

delete[] game\_list;

}

if (next)

{

delete next;

}

}

1. **Листинг Shop.h**

#pragma once

#include "Chain.h"

class Shop : public Сhain {

protected:

string filename;

void result(Game\* game);

public:

Shop(); //конструктор

string get\_filename();

};

1. **Листинг Shop.cpp**

#include "Header.h"

#include "Shop.h"

void Shop::result(Game \* game)

{

cout << "ТОВАР НАЙДЕН В МАГАЗИНЕ" << endl;

}

Shop::Shop()

{

filename = "Shop.txt";

cout << endl << "~~~~~~~~~ СОДЕРЖИМОЕ МАГАЗИНА ~~~~~~~~~" << endl << endl;

read\_from\_file(filename); //заполнение массива с играми в базовом классе

}

string Shop::get\_filename() { return filename; }

1. **Листинг Site.h**

#pragma once

#include "Chain.h"

class Site : public Сhain {

protected:

string filename;

void result(Game\* game);

public:

Site(); //конструктор

string get\_filename();

};

1. **Листинг Site.cpp**

#include "Header.h"

#include "Site.h"

void Site::result(Game \* game)

{

{

cout << "ТОВАР НАЙДЕН НА САЙТЕ" << endl;

}

}

Site::Site()

{

filename = "Site.txt";

cout << endl << "~~~~~~~~~ СОДЕРЖИМОЕ САЙТА ~~~~~~~~~" << endl << endl;

read\_from\_file(filename); //заполнение массива с играми в базовом классе

}

string Site::get\_filename() { return filename; }

1. **Листинг Warehose.h**

#pragma once

#include "Chain.h"

class Warehouse : public Сhain {

protected:

string filename;

void result(Game\* game);

public:

Warehouse(); //конструктор

string get\_filename();

};

1. **Листинг Warehose.cpp**

#include "Header.h"

#include "Warehose.h"

void Warehouse::result(Game \* game)

{

cout << "ТОВАР НАЙДЕН НА СКЛАДЕ" << endl;

}

Warehouse::Warehouse()

{

filename = "Warehouse.txt";

cout << endl << "~~~~~~~~~ СОДЕРЖИМОЕ СКЛАДА ~~~~~~~~~" << endl << endl;

read\_from\_file(filename); //заполнение массива с играми в базовом классе

}

string Warehouse::get\_filename() { return filename; }

1. **Листинг main.cpp**

#include "Header.h"

#include "Game.h"

#include "Chain.h"

#include "Site.h"

#include "Shop.h"

#include "Warehose.h"

const int NotUsed = system("color 1E");

using namespace std;

int main()

{

setlocale(LC\_ALL, "Russian");

Сhain \*chain;

Game \*game;

Site \*site;

Shop \*shop;

Warehouse \*warehouse;

int swtch1 = 1;

while (swtch1 != 0) {

cout << endl << endl << "Выберите действие:" << endl <<

"|1| <-> Вывести все данные из всех источников" << endl <<

"|2| <-> Добавление элементов в базу" << endl <<

"|3| <-> Найти игру на сайте -> магазине -> складе" << endl <<

"|0| <-> Выход" << endl;

cin >> swtch1;

switch (swtch1)

{

case 1: /\*Вывести все данные на экран\*/

chain = new Site;

chain = new Shop;

chain = new Warehouse;

delete chain;

break;

case 2:

cout << "Выберите действие:" << endl <<

"|1| <-> Добавление игр на сайт" << endl <<

"|2| <-> Добавление игр в магазин" << endl <<

"|3| <-> Добавление игр на склад" << endl <<

"|0| <-> Отмена" << endl;

cin >> swtch1;

switch (swtch1)

{

case 1: /\*Добавление игр на сайт\*/

site = new Site;

site->add\_to\_file(site->get\_filename());

delete site;

break;

case 2: /\*Добавление игр в магазин\*/

shop = new Shop;

shop->add\_to\_file(shop->get\_filename());

delete shop;

break;

case 3: /\*Добавление игр на склад\*/

warehouse = new Warehouse;

warehouse->add\_to\_file(warehouse->get\_filename());

delete warehouse;

break;

default:

break;

}

break;

case 3:

std::cout << "OUTPUT:" << std::endl;

chain = new Site; // низший уровень в цепи сайт->магазин->склад

chain->setNext(new Shop)->setNext(new Warehouse);

cout << endl << "Введите данные об игре для поиска " << endl;

game = new Game;

game->set\_game();

chain->search(game);

delete game;

break;

case 0:

break;

default:

break;

}

}

system("pause");

return 0;

}

1. **Листинг Header.h**

#pragma once

#define \_CRT\_SECURE\_NO\_WARNINGS

#include <iostream>

#include <fstream>

#include <string>

using namespace std;

**Приложение №3 Примеры**

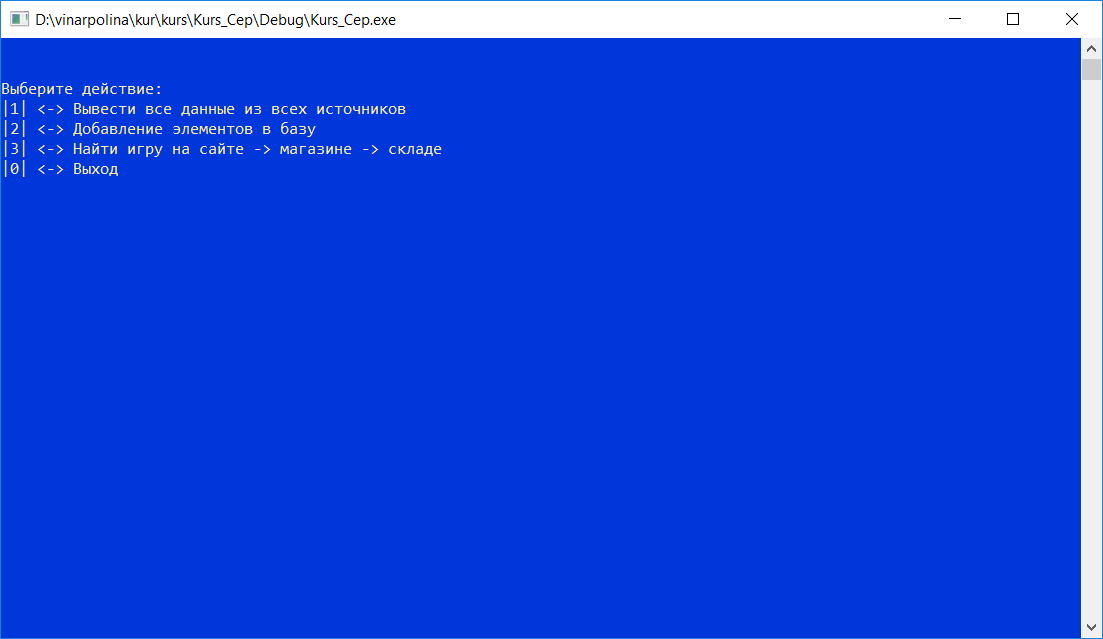


Рисунок 1

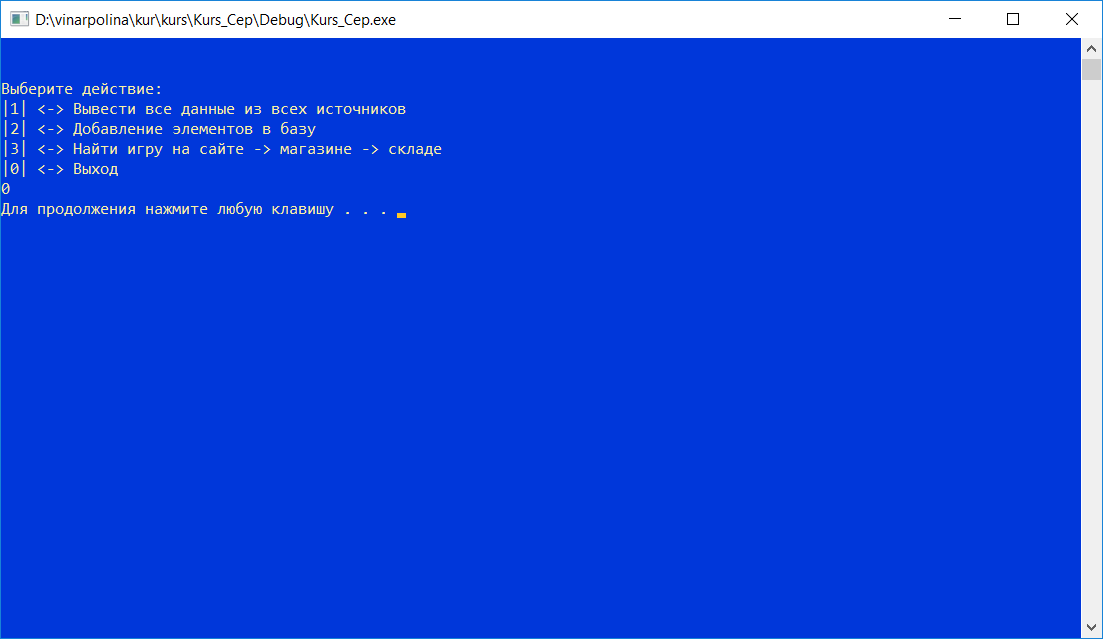


Рисунок 2

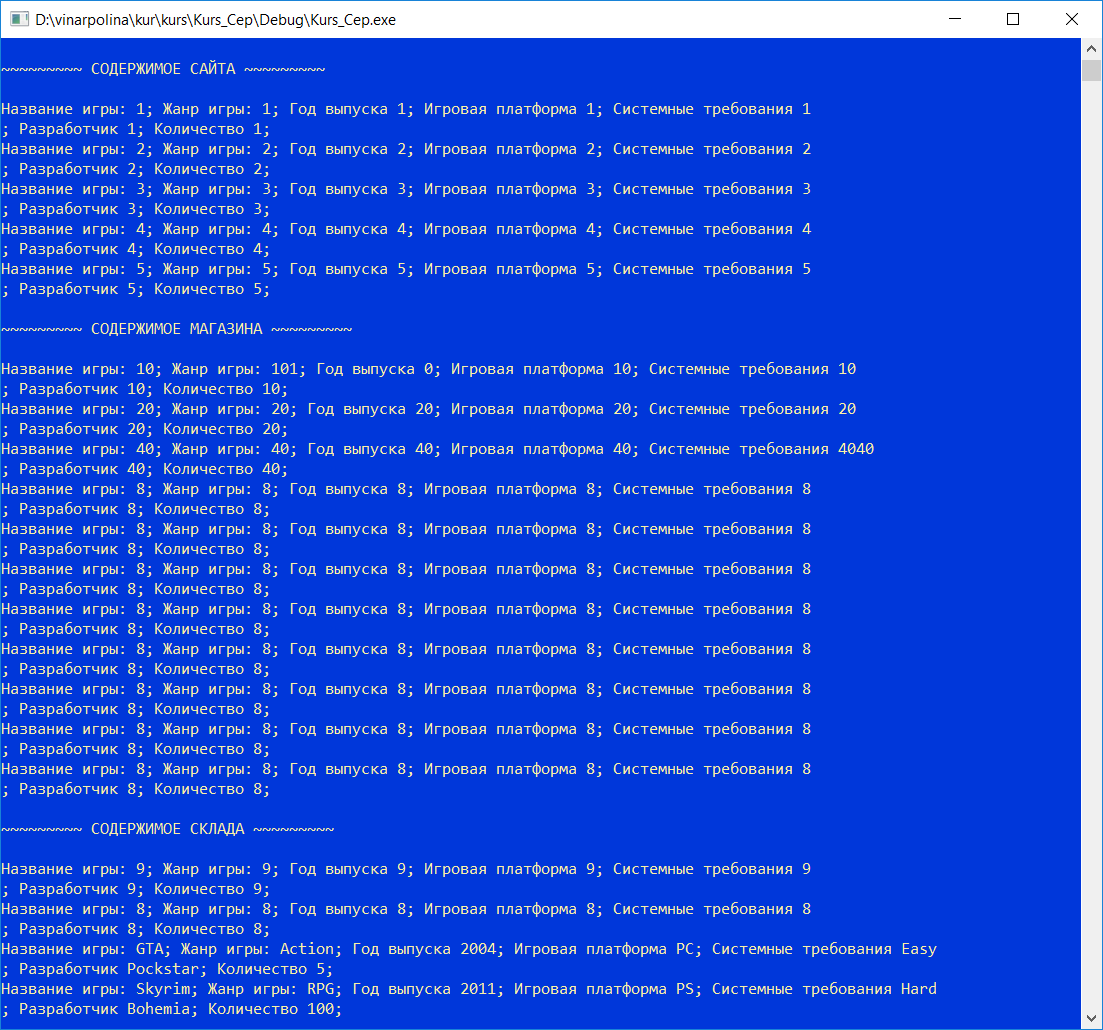


Рисунок 3

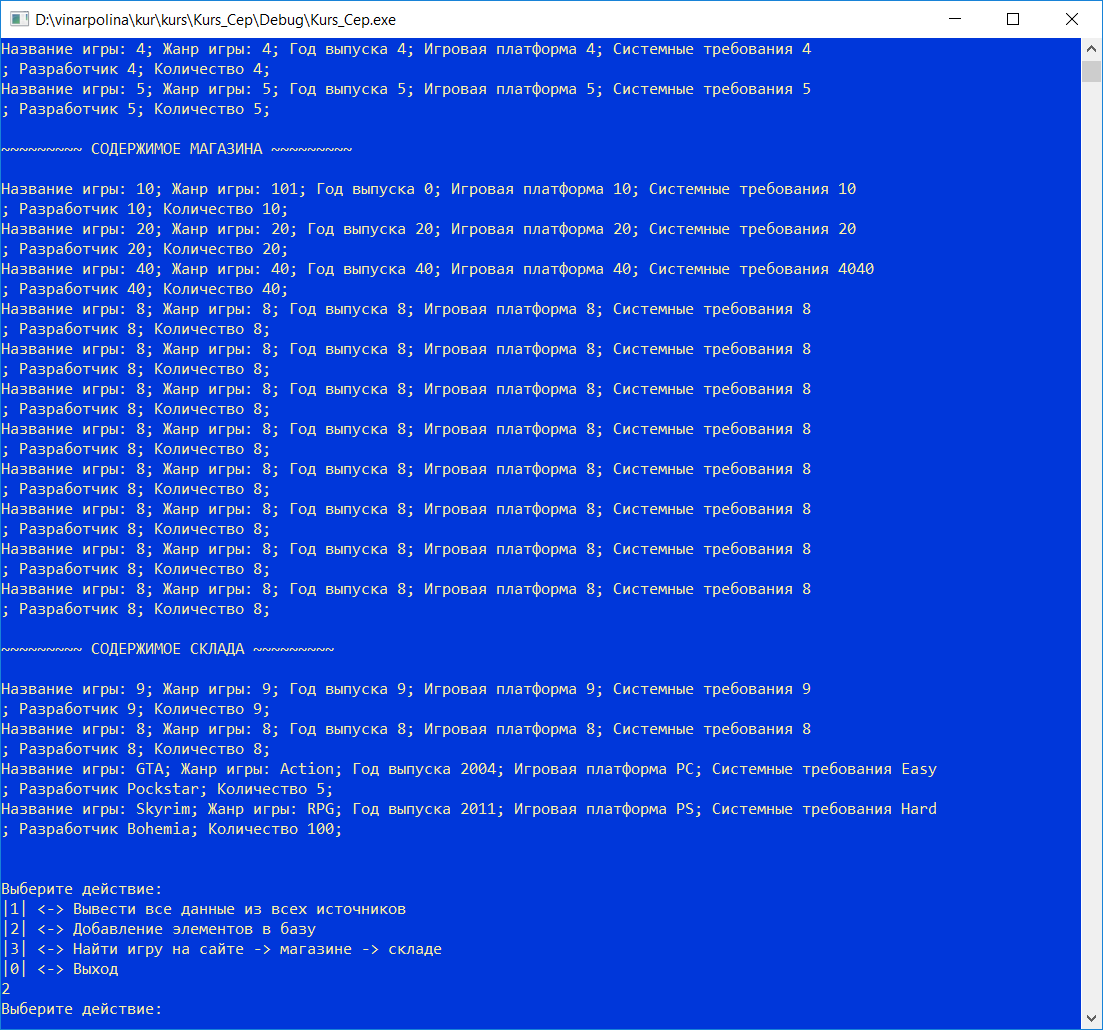
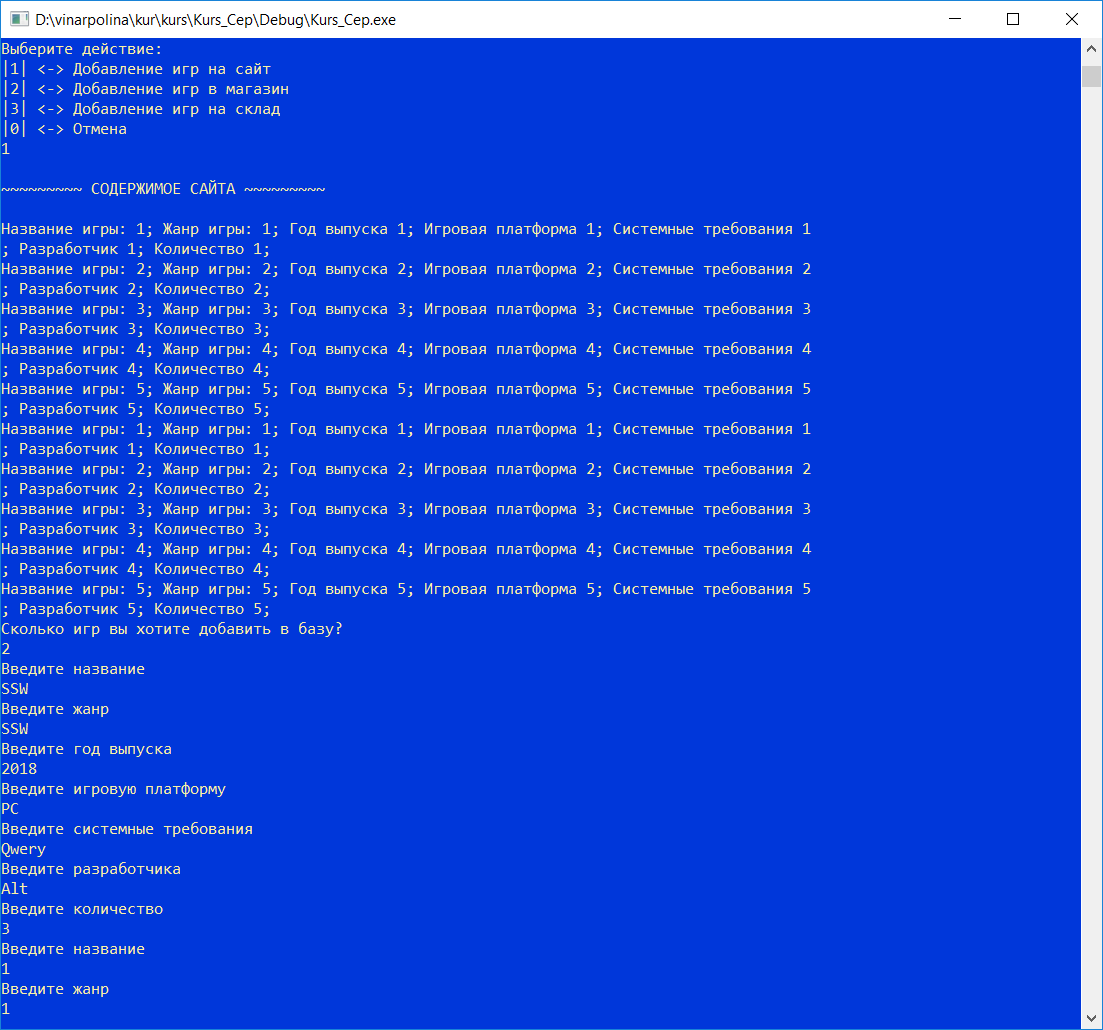


Рисунок 4



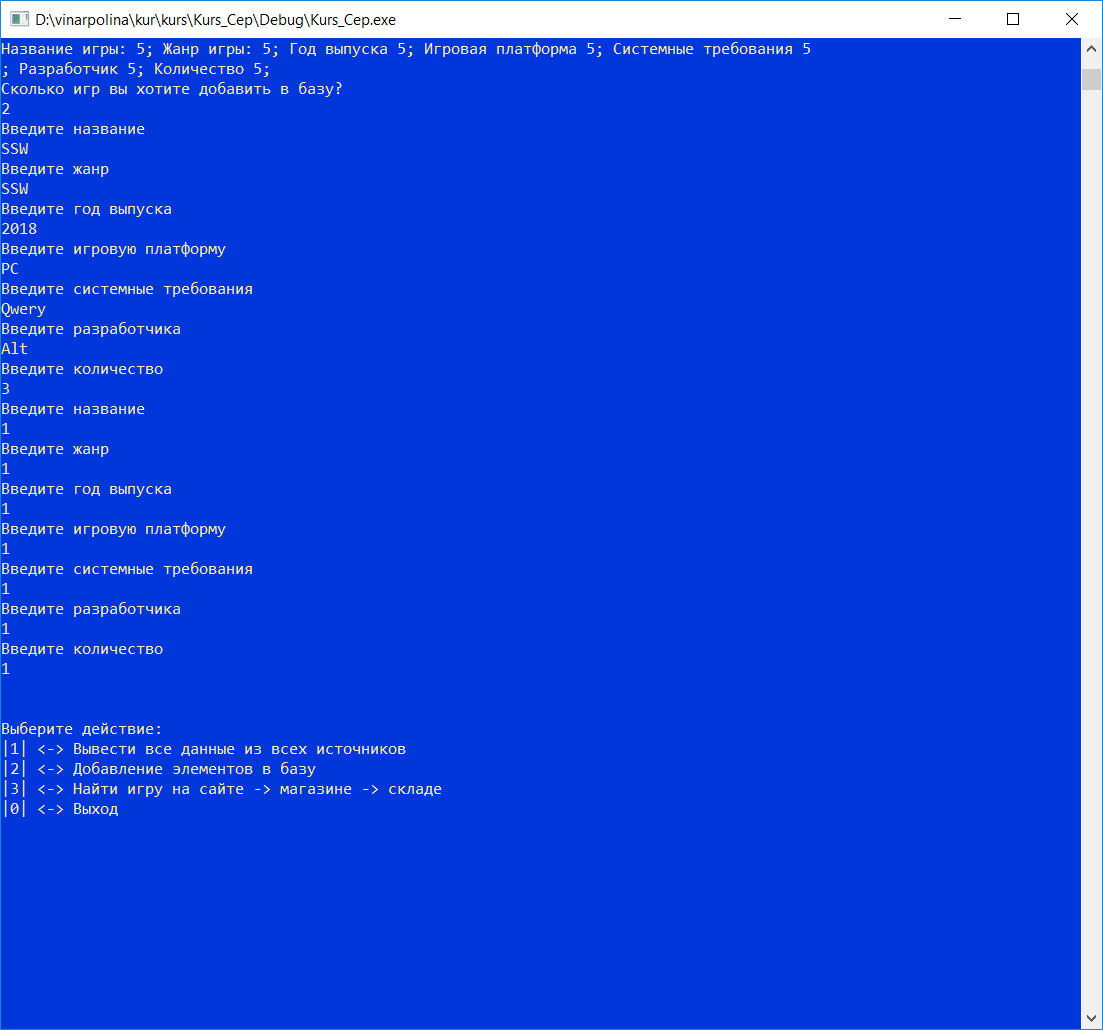
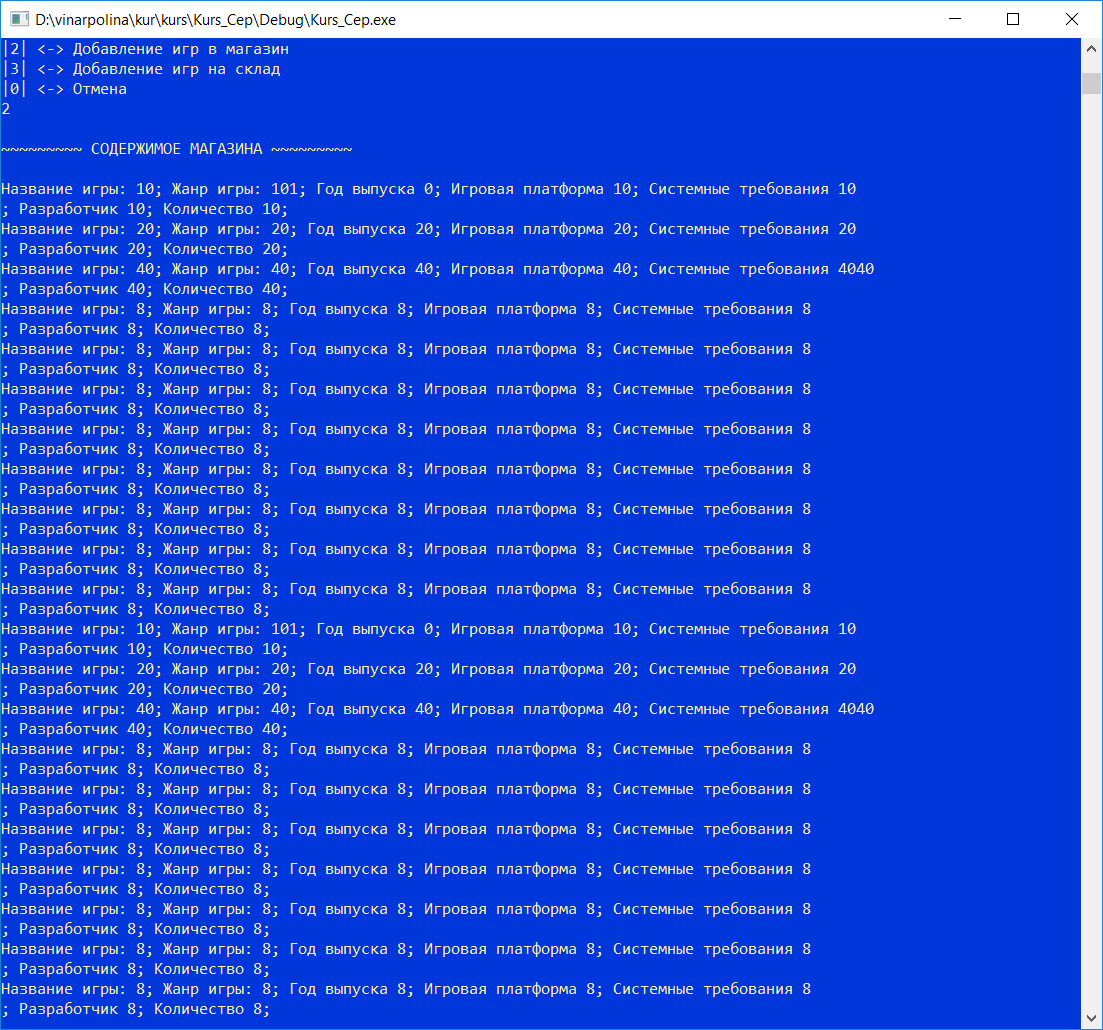


Рисунок 5



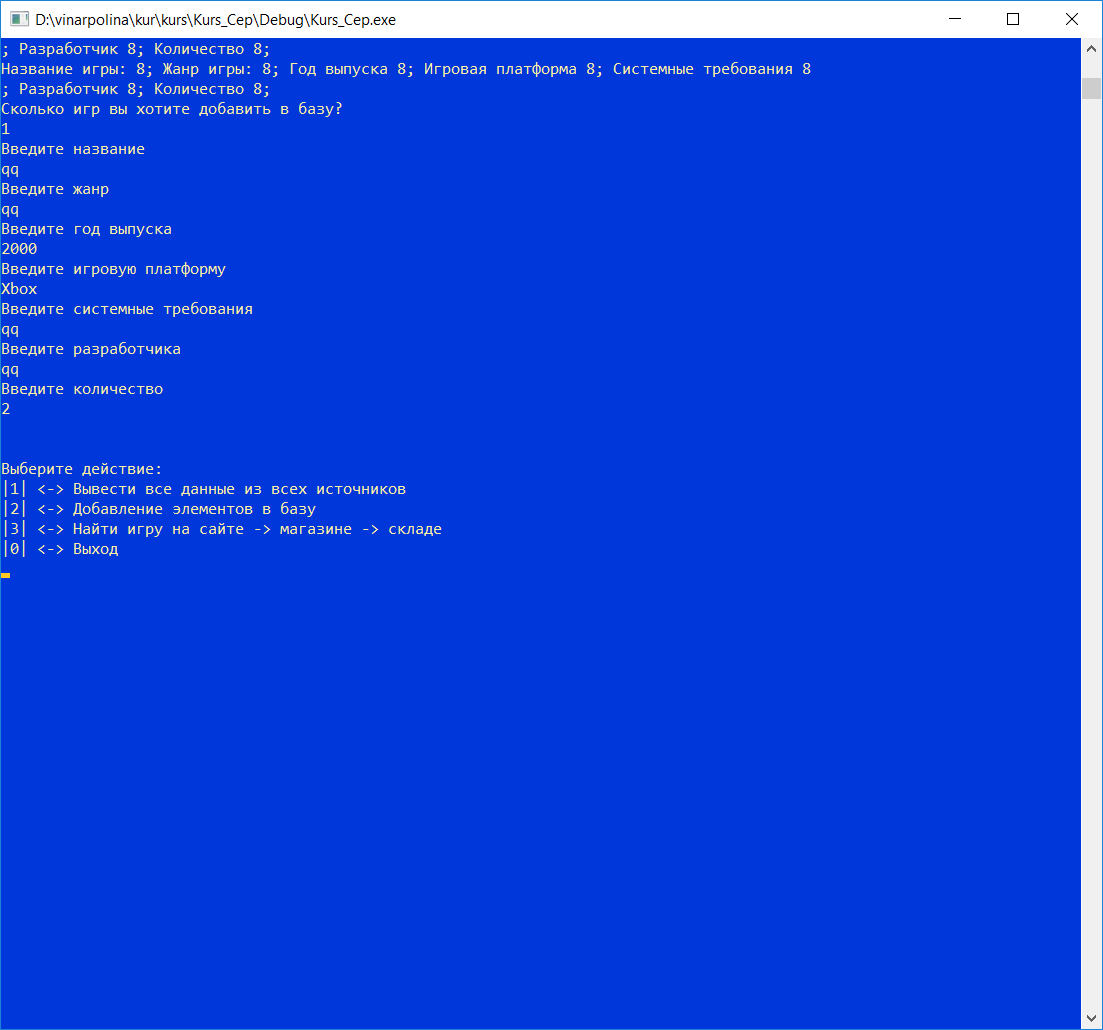


Рисунок 6

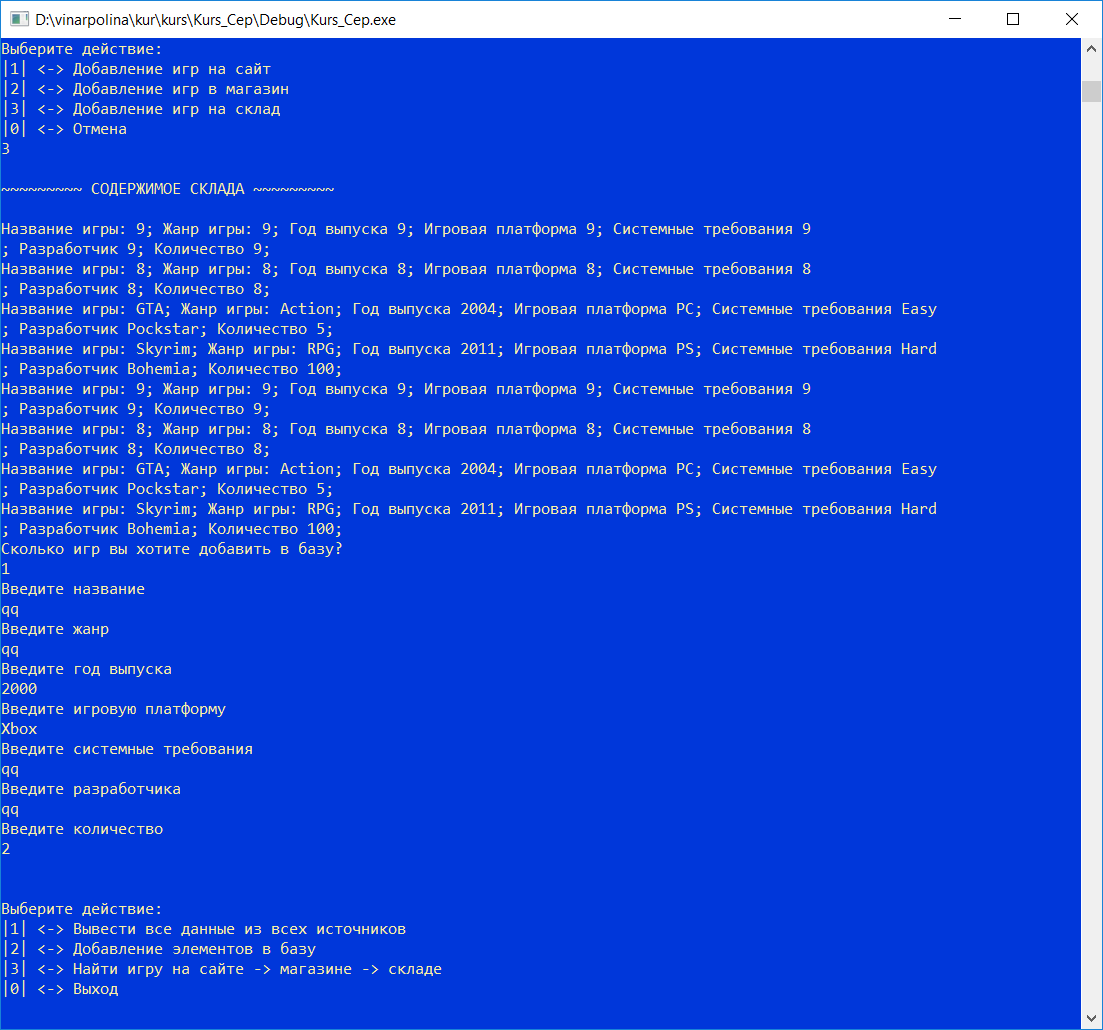
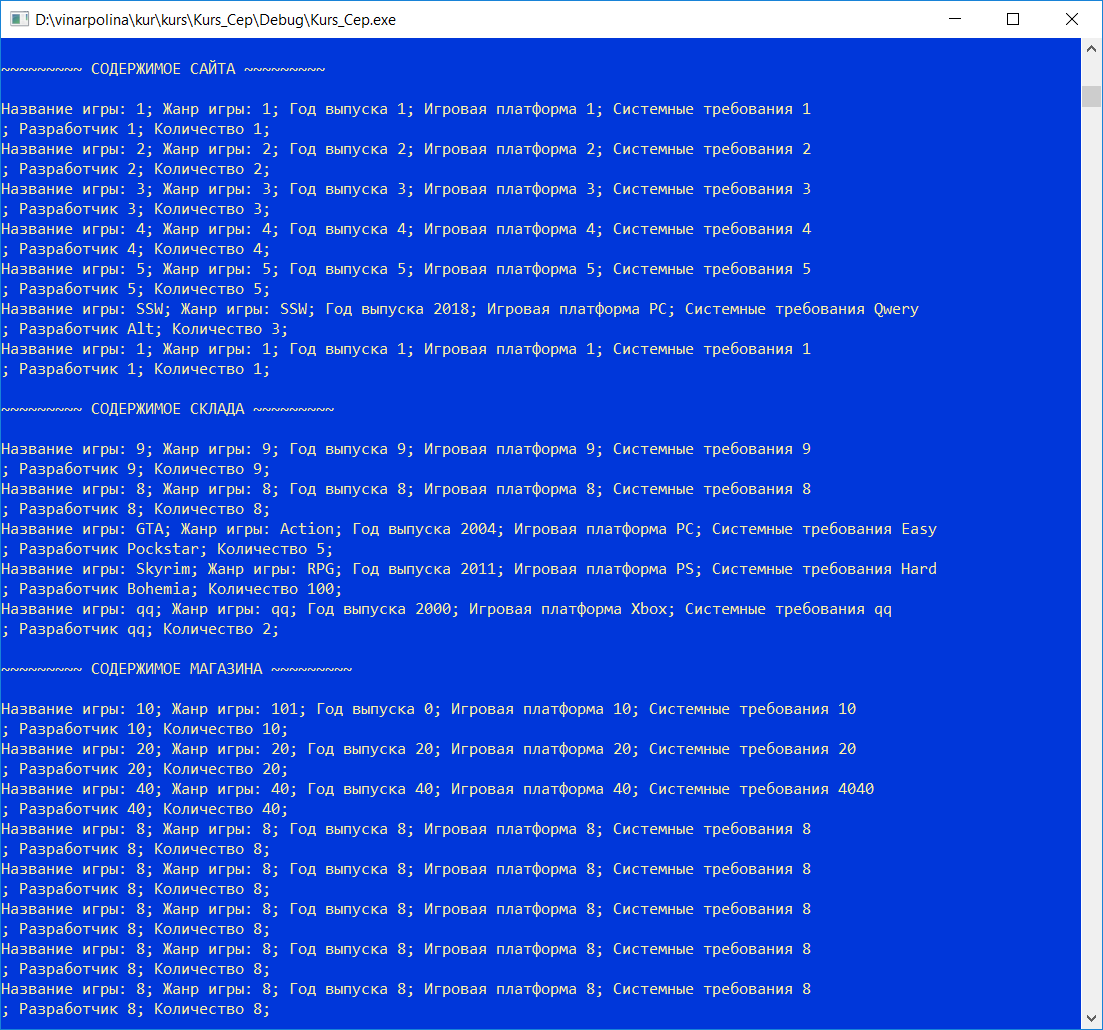


Рисунок 7



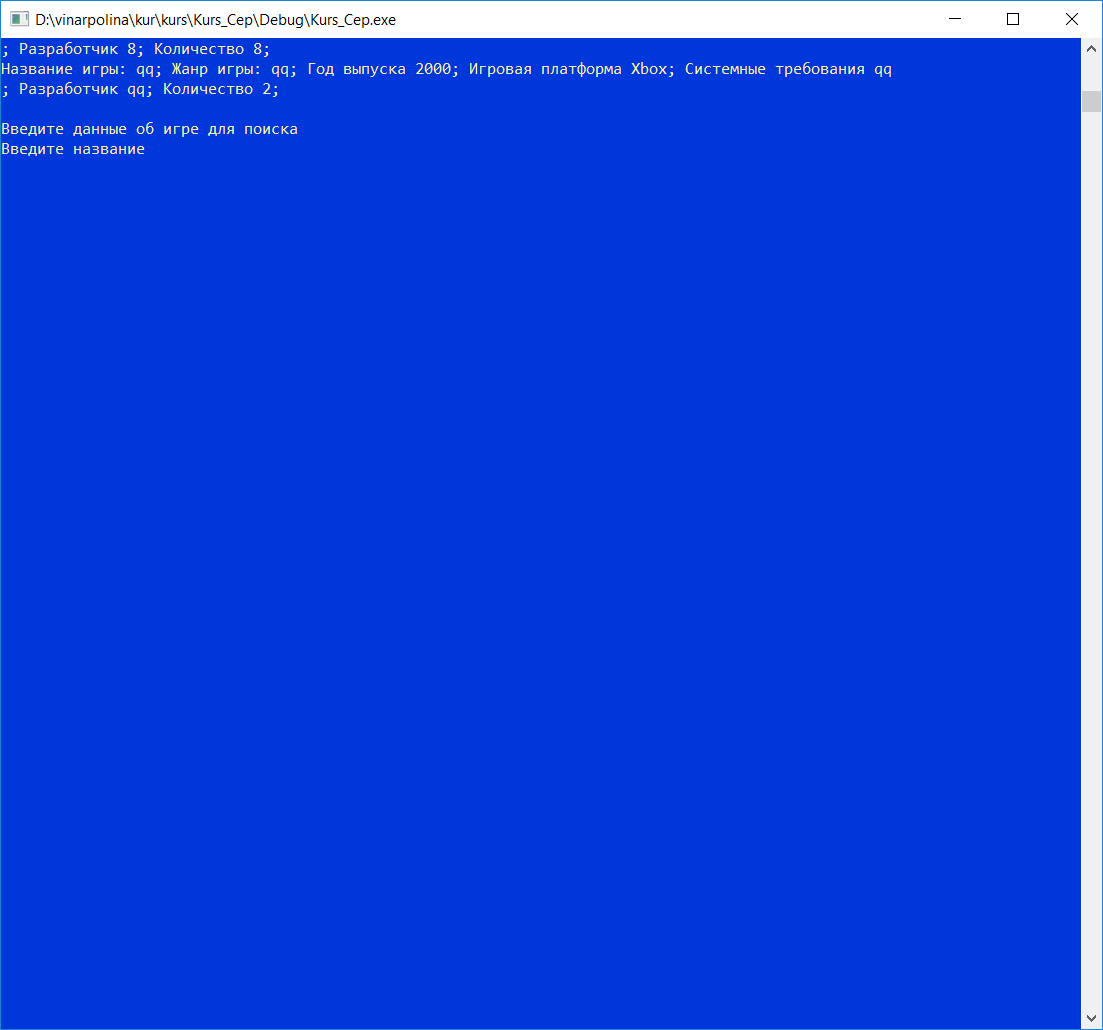


Рисунок 8

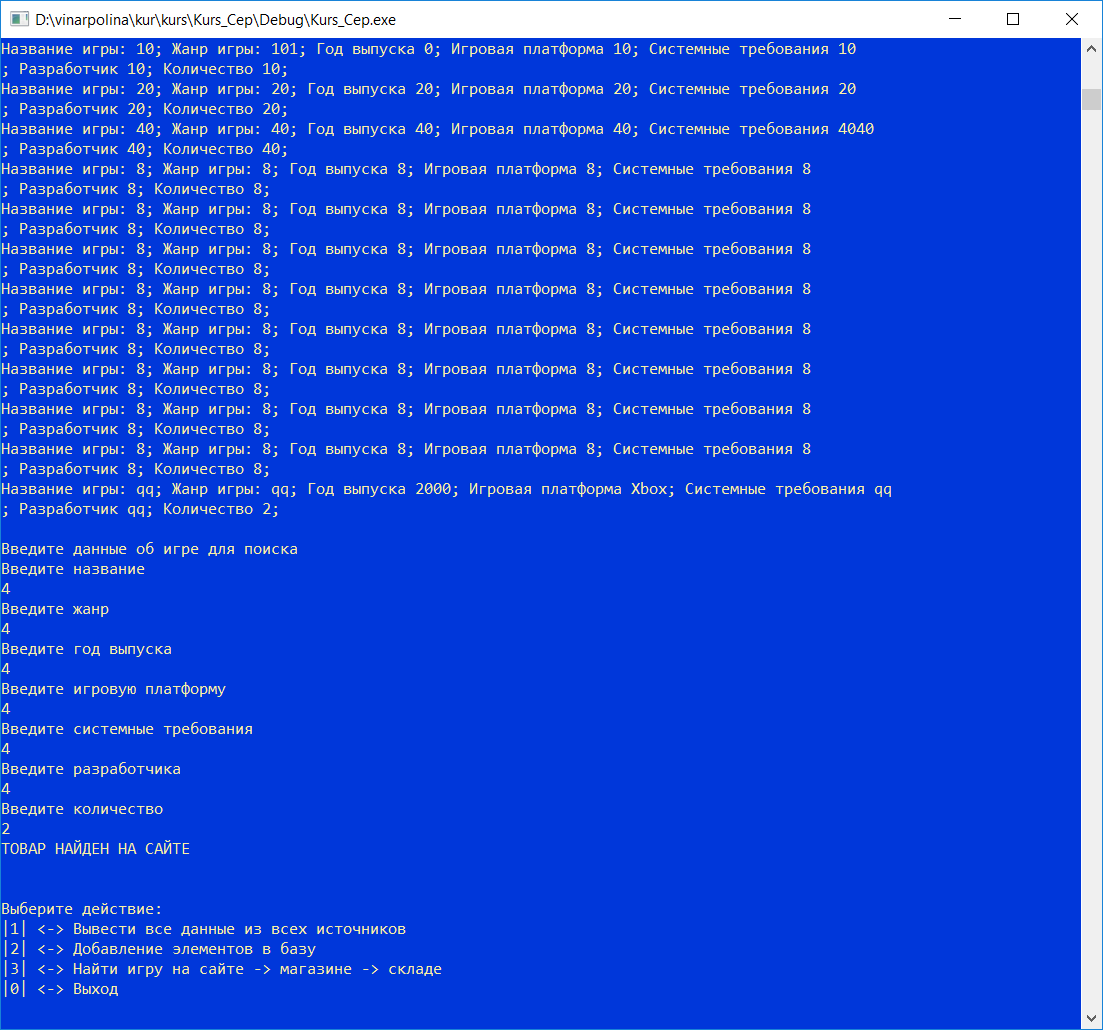


Рисунок 9

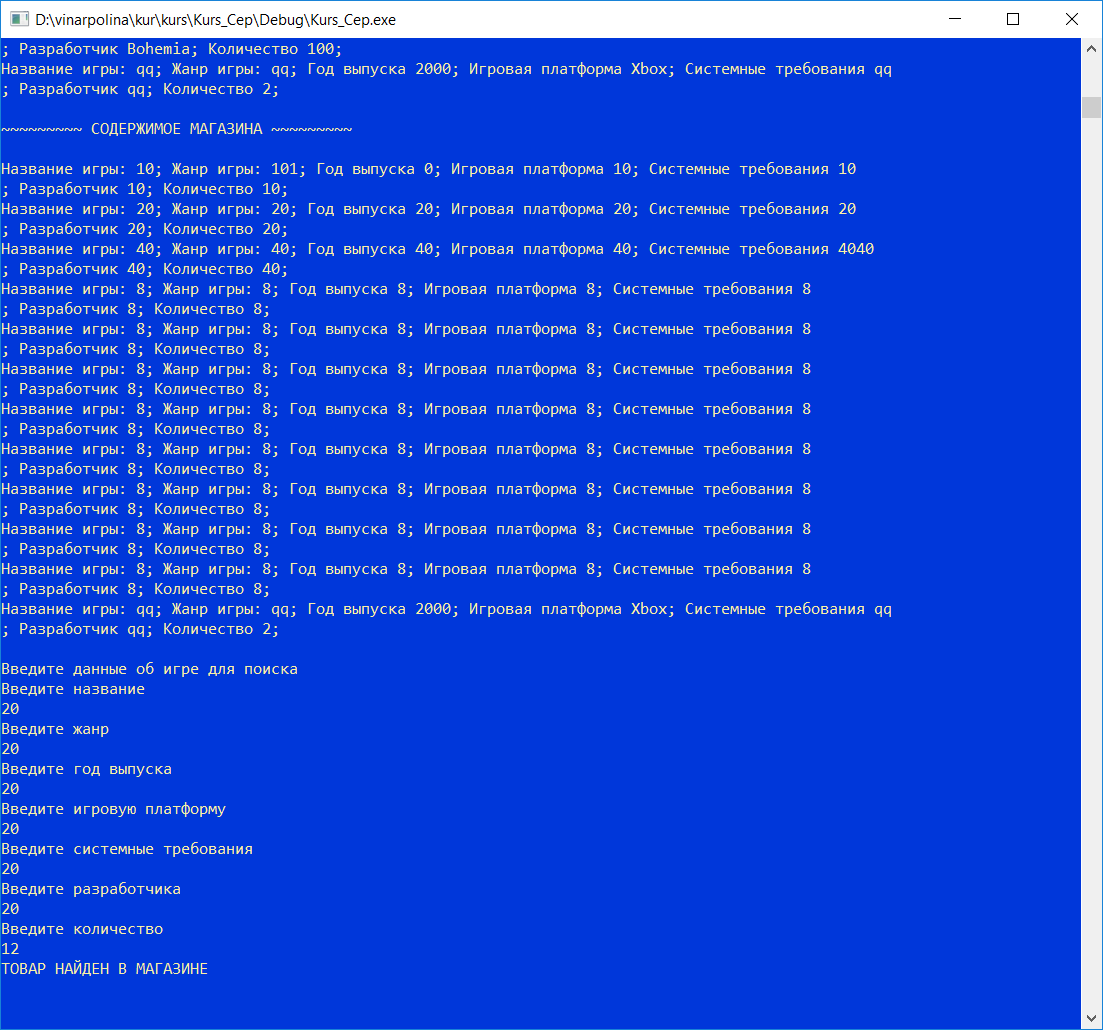


Рисунок 10

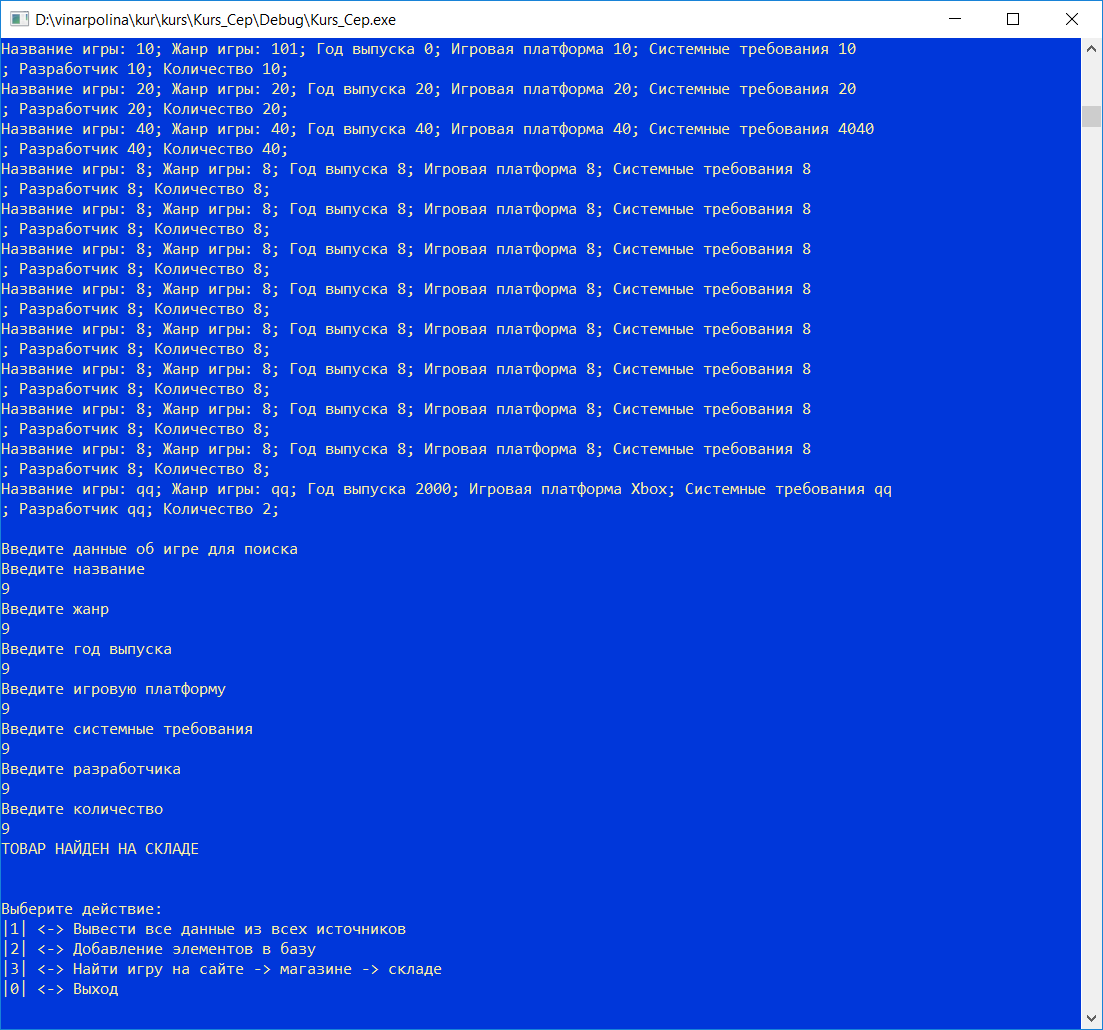


Рисунок 11

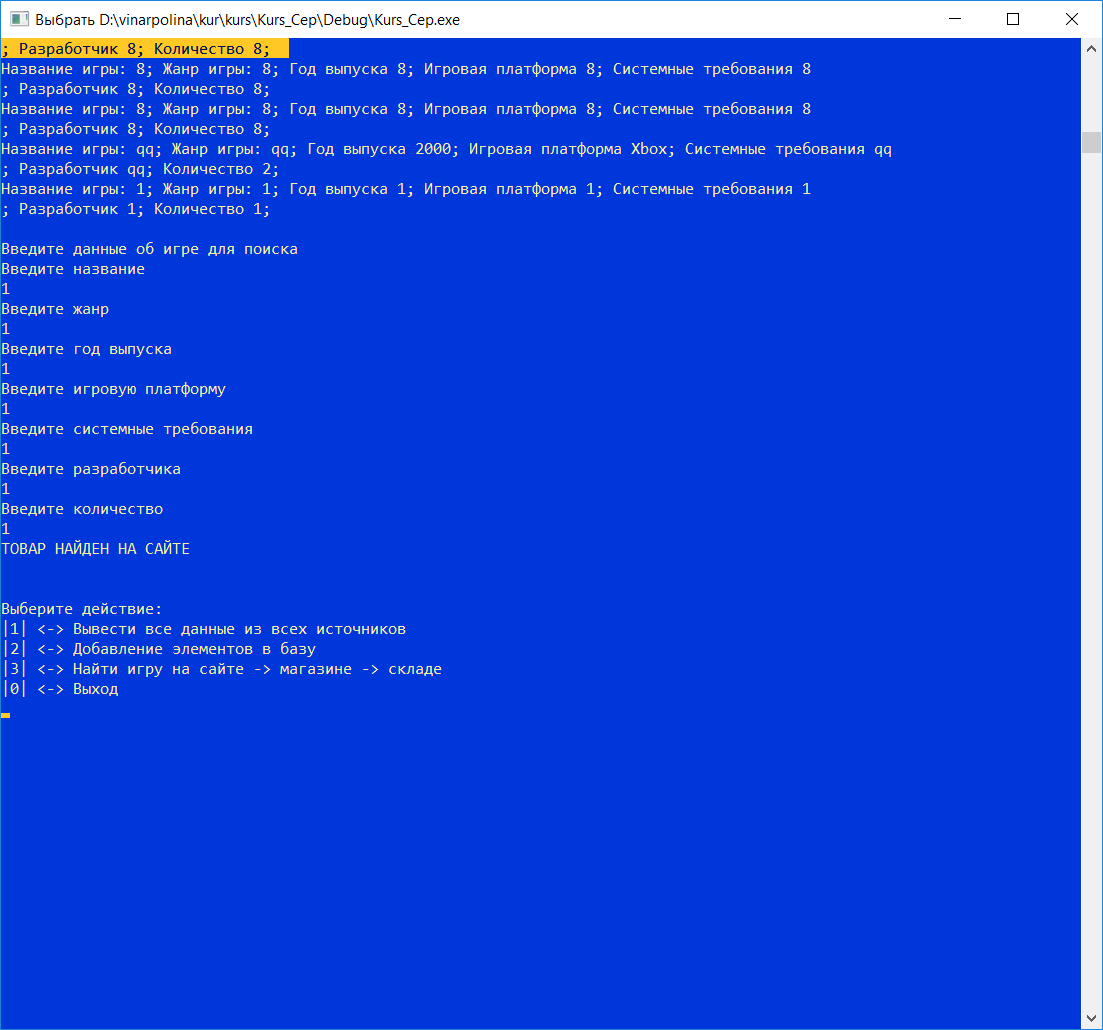


Рисунок 12

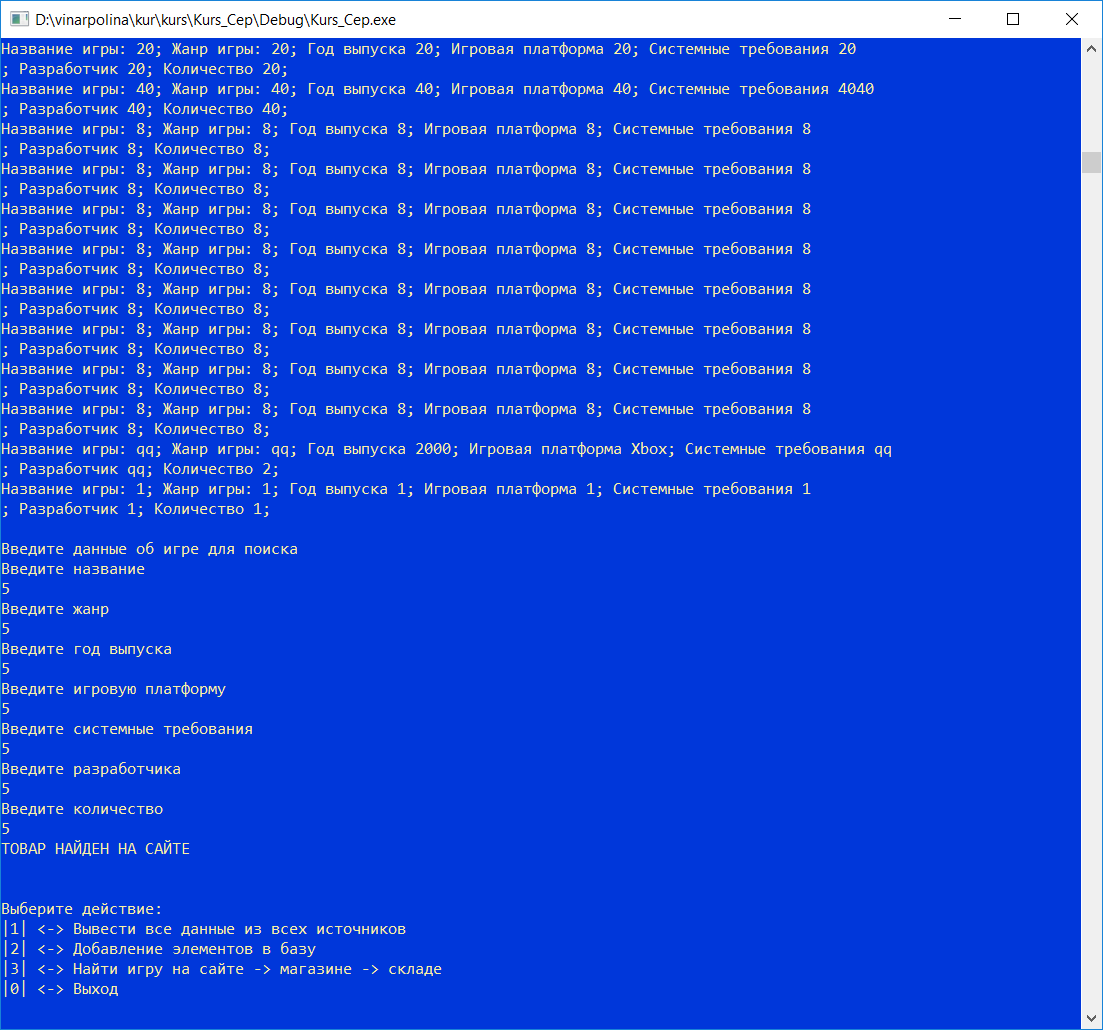


Рисунок 13

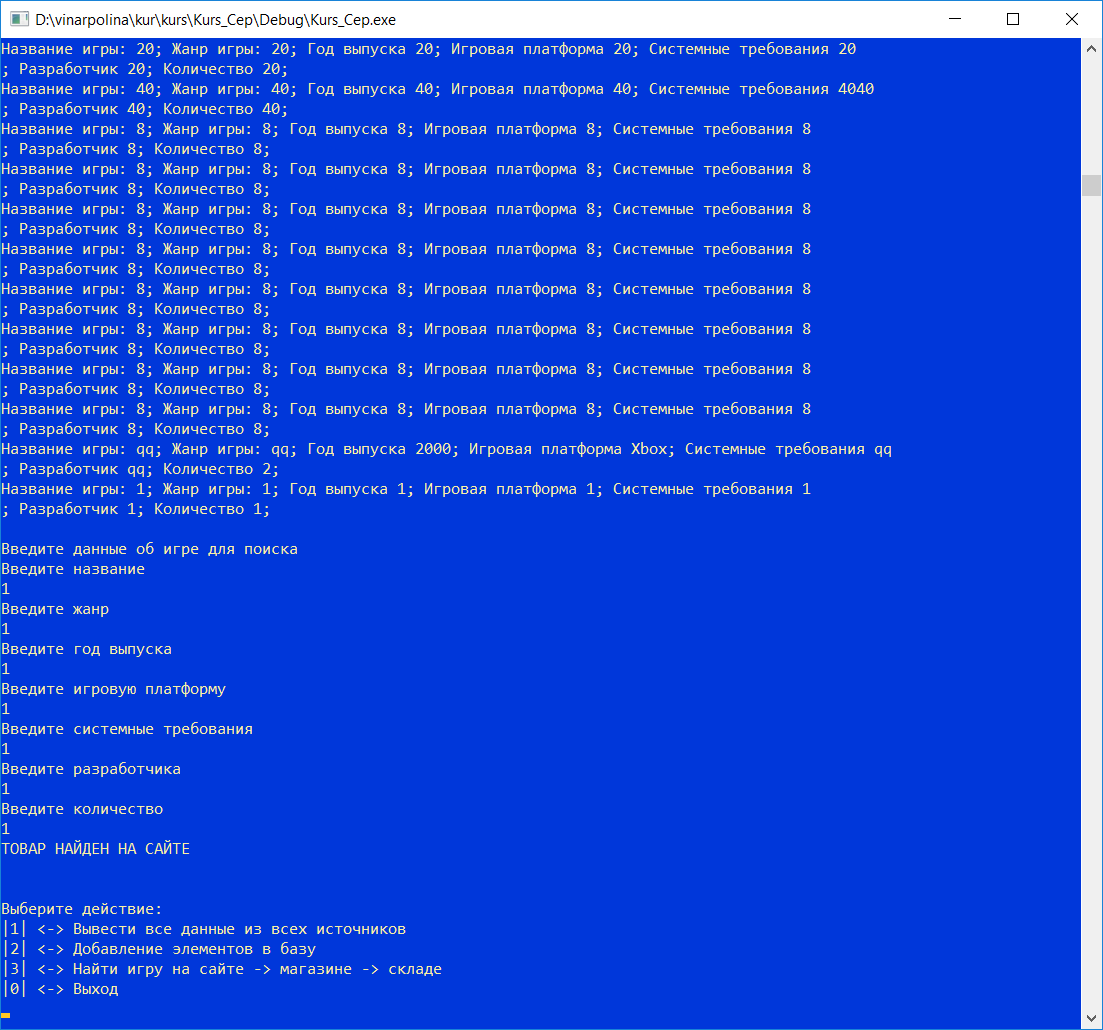


Рисунок 14

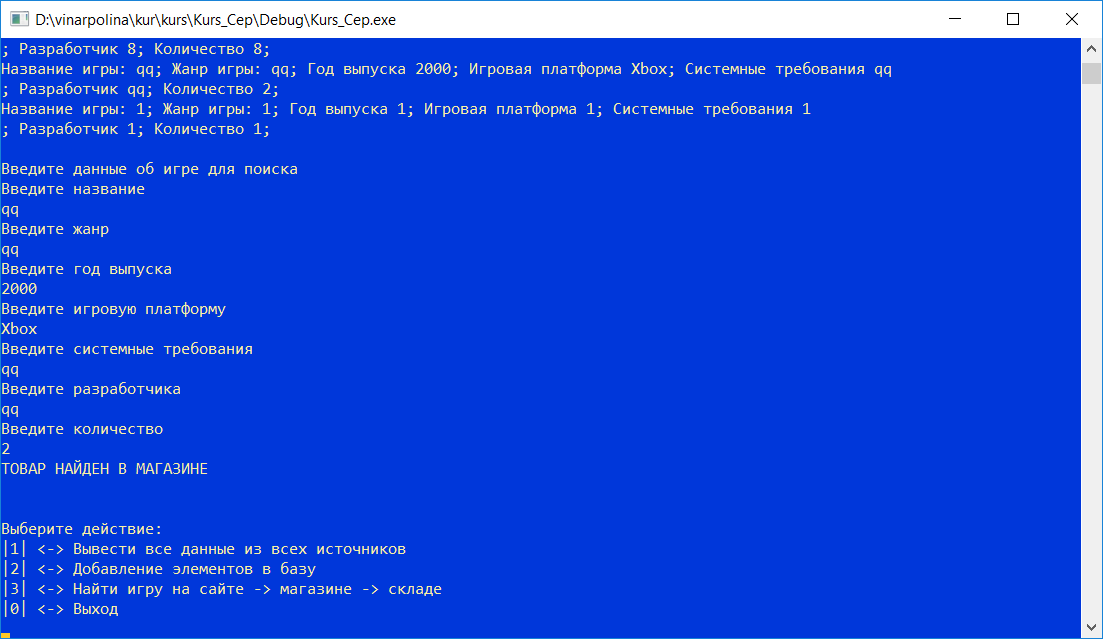


Рисунок 15

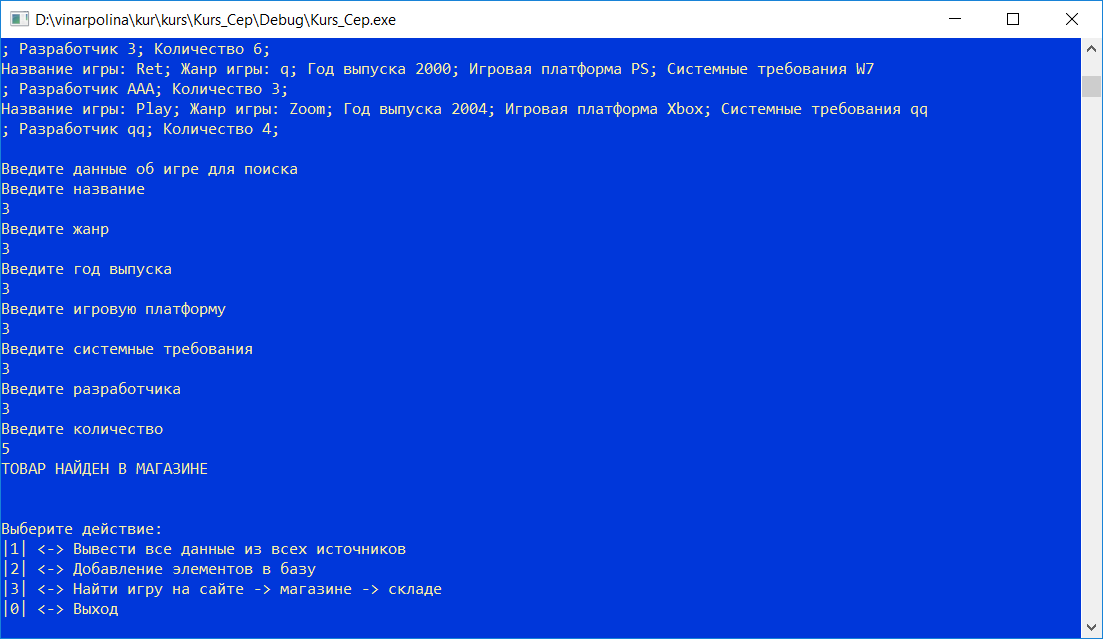


Рисунок 16

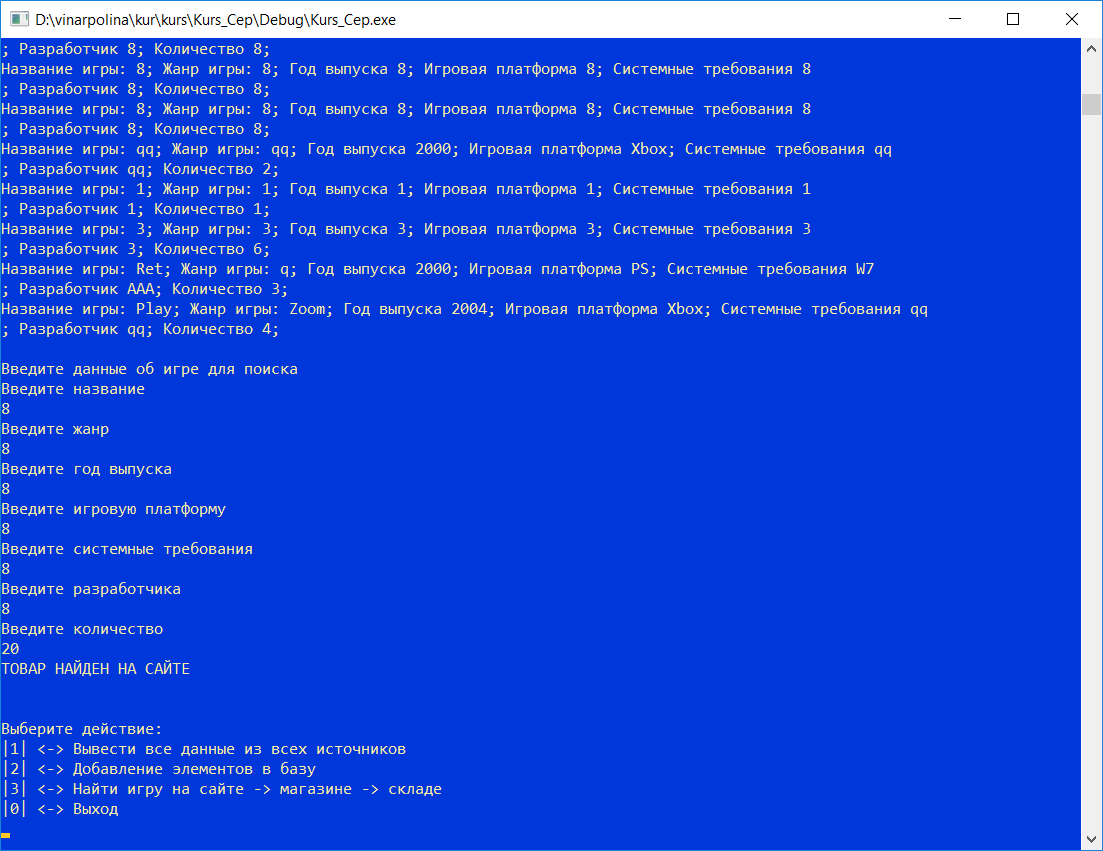


Рисунок 17

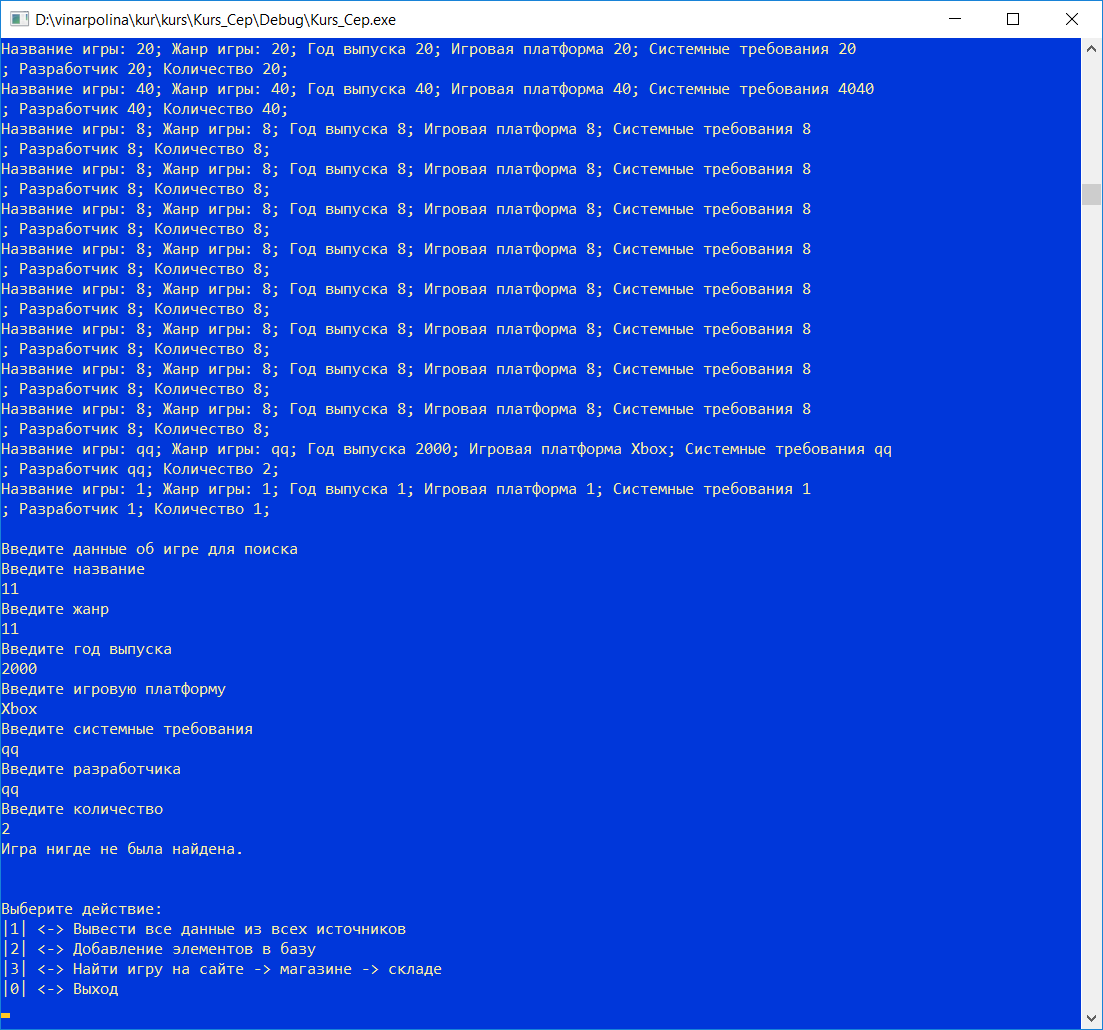


Рисунок 18