Министерство образования Республики Беларусь

Учреждение образования «Белорусский государственный университет   
информатики и радиоэлектроники»

Военный Факультет

Кафедра электронных вычислительных машин

Дисциплина: Программирование на языках высокого уровня

ПОЯСНИТЕЛЬНАЯ ЗАПИСКА

к курсовому проекту

на тему

«Информационная система по продаже компьютерных комплектующих»

БГУИР КП 1-40 02 01 112 ПЗ

Студент: гр. 130501 Новиков Н. Д.

Руководитель: Ковальчук А. М.

Минск 2022

Учреждение образования

«Белорусский государственный университет информатики   
и радиоэлектроники»

Военный факультет

УТВЕРЖДАЮ

Заведующий кафедрой

*.*

(подпись)

2022 г.

ЗАДАНИЕ

по курсовому проектированию

Студенту *Новикову Никите Дмитриевичу*

1. Тема проектаИнформационная система по продаже компьютерных комплектующих

2. Срок сдачи студентом законченного проекта *15 декабря 2022 г.*

3. Исходные данные к проекту *текстовые файлы, в которых содержится информация:Сенсорные мыши (*Год выпуска, Бренд, Цена, Чувст. Мыши, Материал, Частота курсора, Кол-во датчиков*),Беспроводные мыши (*Год выпуска, Бренд, Цена, Чувст. Мыши, Материал, Способ прокрутки, Длину провода, Время отклика*),LCD-мониторы(*Год выпуска, Бренд, Цена, Разреш. Экрана, Кол-во дюймов, Mатрица, Время вкл.*),PDP-мониторы(*Год выпуска, Бренд, Цена, Разреш. Экрана, Кол-во дюймов, Плазма, Яркость*),Контейнер: Deque; Итератор:* *Iter , стандартные контейнеры: list, vector;своя реализация string : класс String.*

4. Перечень графического материала (с точным обозначением обязательных чертежей и графиков)

*1. Диаграмма классов*

*2. Схема алгоритма метода:––––––––––––––––––––––––––––––*

template<class T>

Deque<T>& Algorithm<T>::search(Node<T>\* beg, T \_obj)

*3. Схема алгоритма метода:––––––––––––––––––––––––––––––   –*

template<class T>

void Algorithm<T>::sort(Deque<T>& \_a)

4. Консультант по проекту *Ковальчук А. М*.

5. Дата выдачи задания *15 сентября 2022 г.*

6. Календарный график работы над проектом на весь период проектирования (с обозначением сроков выполнения и трудоемкости отдельных этапов):

*разделы 1, 2 к 1 октября 2022 г. – 20 %;*

*разделы 3, 4 к 1 ноября 2022 г. – 30 %;*

*разделы 5, 6 ,7 к 1 декабря 2022 г. – 30 %;*

*оформление пояснительной записки и графического материала к 14 декабря 2022 г. – 20 %*

*Защита курсового проекта с 21 декабря 2022 г. по 28 декабря 2022 г.*

РУКОВОДИТЕЛЬ– –––––––––*Ковальчук А. М.*

Задание принял к исполнению –Х *Новиков Н.Д.*

СОДЕРЖАНИЕ

[ВВЕДЕНИЕ 6](#_Toc121470076)

[1 ПОСТАНОВКА ЗАДАЧИ 8](#_Toc121470077)

[2 ФУНКЦИОНАЛЬНОЕ ПРОЕКТИРОВАНИЕ 9](#_Toc121470079)

[2.1 Блок приложений 10](#_Toc121470080)

[2.1.1 Блок выбора типа объекта 10](#_Toc121470080)

[2.1.2 Блок выбора метода для работы с объектом 10](#_Toc121470080)

[2.1.3 Блок выбора метода для работы с объектом 10](#_Toc121470080)

[2.2 Текстовые файлы 11](#_Toc121470084)

[2.2.1 Файл TouchMouse.txt 12](#_Toc121777988)

[2.2.2 Файл WiredMouse.txt 13](#_Toc121777989)

[2.2.3 Файл LCD.txt 13](#_Toc121777990)

[2.2.4 Файл PDP.txt 14](#_Toc121777991)

[2.3 Шаблон программы 14](#_Toc121777991)

[2.4 Класс Итератор 15](#_Toc121470085)

[2.5 Свой класс string 17](#_Toc121470085)

[3 РАЗРАБОТКА ПРОГРАММНЫХ МОДУЛЕЙ 19](#_Toc121470086)

[3.1 Алгоритм по шагам 19](#_Toc121470087)

[3.2 Схемы алгоритмов 20](#_Toc121470087)

[4 ТЕСТИРОВАНИЕ 21](#_Toc121470088)

[5 СПИСОК ЛИТЕРАТУРЫ 29](#_Toc121470088)

[ЗАКЛЮЧЕНИЕ 30](#_Toc121470090)

[Приложение А 31](#_Toc121470091)

[Приложение Б 32](#_Toc121470092)

[Приложение В 33](#_Toc121470093)

[Приложение Г 34](#_Toc121470094)

[Приложение Д 35](#_Toc121470095)

# **ВВЕДЕНИЕ**

Объектно-ориентированное программирование представляет собой технологию программирования, которая базируется на классификации и абстракции объектов. Одним из наиболее популярных средств объектно-ориентированного программирования, позволяющим разрабатывать программы, эффективные по объёму кода и скорости выполнения является С++.

C++ — мощный язык, содержащий средства создания эффективных программ практически любого назначения, от низкоуровневых утилит и драйверов до сложных программных комплексов самого различного назначения. В частности:

- предсказуемое выполнение программ является важным достоинством для построения систем реального времени. Весь код, неявно генерируемый компилятором для реализации языковых возможностей (например, при преобразовании переменной к другому типу), определён в стандарте. Строго определены места программы, в которых он выполняется. Это даёт возможность замерять или рассчитывать время реакции программы на внешнее событие.

- автоматический вызов деструкторов объектов при их уничтожении, в порядке, обратном вызову конструкторов, что позволяет избежать утечек памяти.

- язык поддерживает понятие (const) константности, что позволяет компилятору, например, диагностировать ошибочные попытки изменения значения константной переменной. Объявление константности даёт программисту, читающему текст программы дополнительное представление о правильном использовании классов и функций, а также может являться подсказкой для оптимизации.

- язык имеет крупную стандартную библиотеку STL, которая включает в себя распространённые контейнеры и алгоритмы.

- С++ сочетает в себе как возможности низкоуровневых языков программирования, так и возможности высокоуровневых. Имеется возможность работы на низком уровне с памятью, адресами.

- кроссплатформенность: стандарт языка накладывает минимальные требования на ЭВМ для запуска скомпилированных программ. Доступны компиляторы для большого количества платформ, на языке C++ разрабатывают программы для самых различных платформ и систем.

- эффективность. Язык спроектирован так, чтобы дать программисту максимальный контроль над всеми аспектами структуры и порядка исполнения программы. Ни одна из языковых возможностей, приводящая к дополнительным накладным расходам, не является обязательной для использования — при необходимости язык позволяет обеспечить максимальную эффективность программы.

# **1 ПОСТАНОВКА ЗАДАЧИ**

Темой данного курсового проекта является «Информационная система по продаже компьютерных комплектующих».

Данная сиcтема содержит программное обеспечение для персонального компьютера.

*Цели* курсового проектирования: овладеть практическими навыками проектирования и разработки законченного, отлаженного и протестированного программного продукта с использованием языка высокого уровня С++;

Минимальные требования к реализации программного продукта:

* хранение пользовательских данных
* запись, удаление и изменение информации
* простой и понятный интерфейс.

*Задачами* курсового проекта (работы) как этапа подготовки к дипломному проектированию являются:

– освоение, углубление и обобщение знаний, полученных студентами в процессе обучения;

– приобретение практических навыков и развитие творческих подходов к решению конкретной инженерной или инженерно-экономической задачи;

– формирование умений использовать справочную литературу, нормативную, правовую, нормативно-техническую документацию, осуществлять патентный поиск;

– приобретение навыков по оформлению текстовой и графической документации согласно требованиям государственных стандартов и стандарта предприятия Дипломные проекты (работы).

# **2 ФУНКЦИОНАЛЬНОЕ ПРОЕКТИРОВАНИЕ**

Основной функцией компьютера является обработка информации.Программная обработка данных на компьютере реализуется следующим образом:

1. После запуска на выполнение программы, хранящейся во внешней долговременной памяти, она загружается в оперативную память.

2. Процессор последовательно считывает команды программы и выполняет их.

3. Необходимые для выполнения команды данные загружаются из внешней памяти в оперативную и над ними производятся необходимые операции. Данные, полученные в процессе выполнения команды, записываются процессором обратно в оперативную или внешнюю память.

4. В процессе выполнения программы процессор может запрашивать данные с устройств ввода информации и пересылать данные на устройства вывода информации.

Программная конфигурация ПК многоуровневая. Это связано с тем, что требования к программам, предназначенным для работы с устройствами, существенно отличаются от требований к программам, предназначенным для работы с людьми. Общий принцип такой: чем ниже уровень программ, тем больше они работают с устройствами и меньше с человеком. Этот принцип соблюдается во всей компьютерной технике от отдельного ПК до всемирной компьютерной сети Интернет.

Для разработки больших и сложных программ программисту необходимо овладеть специальными приемами получения рациональной структуры программы, которая обеспечивает почти двукратное сокращение объема программирования. Подчиненность модулей программы отражается в схеме, представленной в приложении А. Однако последняя не отражает порядок их вызова или функционирование программы.

# **2.1 Блок приложения**

В приложении можно выделить следующие структурные блоки:

* блок выбора типа объекта;
* блок выбора метода для работы с объектом;
* блок работы с STL;

# **2.1.1 Блок выбора типа объекта**

Данный блок предоставляет пользователю возможность выбора типа объекта для работы с ним в программе. Блок имеет удобный пользовательский интерфейс, в котором находится 4 пункта. В первоначальном окне пользователь может выбрать тип объекта, также может выйти из программы.

# **2.1.2 Блок(ADM) выбора метода для работы с объектом**

Данный блок предоставляет администратору, выбор метода, для работы с объектами. Блок имеет удобный пользовательский интерфейс, в котором находится 14 пунктов. В окне пользователь может выбрать метод, для работы с объектами, также может вернуться к блоку выбора типа объекта.

**2.1.3 Блок(USER) выбора метода для работы с объектом**

Данный блок предоставляет пользователю, выбор метода, для работы с объектами. Блок имеет удобный пользовательский интерфейс, в котором находится 5 пунктов. В окне пользователь может выбрать метод, для работы с объектами, также может вернуться к блоку выбора типа объекта.

# **2.2 Текстовые файлы**

**Текстовой файл** — это файл, компонентами которого являются символьные строки переменной длины, заканчивающиеся специальным маркером конца строки.

Преимущества:

* Минимальный объём файла (при малом количестве текстовых данных).
* Универсальность — текстовый файл может быть прочитан (так или иначе) на любой системе или ОС, особенно если речь идёт об однобайтных кодировках вроде ASCII — они не подвержены многим проблемам, характерным для других форматов файлов — для них не важна разница в порядке байтов или длине машинного слова на разных платформах.
* Устойчивость — каждое слово и символ в таком файле самодостаточны, и если случится повреждение такого файла, то в этом случае обычно легче восстановить данные и продолжить обработку остального содержимого — повреждение любого из байтов такого файла никак не сказывается на сохранность остальных, в отличие от или сжатых или двоичных (напр. \*.doc). Многие распространённые системы управления версиями (например в MS Word) рассчитаны на текстовые файлы, и с двоичными файлами могут работать только как с единым целым — при повреждении даже нескольких байтов такой файл восстановлению не подлежит.

Для работы с файлами я использовал классы ofstream и ifstream.

В таблице 1.1 представлены команды используемые в приложении для работы с файлами.

Таблица 1.1 – Используемые методы fromFile и toFile, checkCount, clear.

| Тип метода | Метод | Описание метода |
| --- | --- | --- |
| Методы доступа к файлу | open() | Этим методом можно открыть заданный файл, сопоставив его с одним из объектов потока. В зависимости от передаваемых аргументов, файл может быть открыт для чтения, для записи (либо для полной, либо для добавления данных), как бинарный, или как текстовый файл. |
|  | close() | Функция закрывает файл, то есть прекращает доступ к нему, таким образом освобождая его для других функций или программ. |

# **2.2.1 Файл TouchMouse.txt**

В файле TouchMouse.txt находится основная информация о сенсорных мышках по физической подготовки. Более подробно рассмотрим поля, который хранятся в данном файле:

* поле yearOfIssue. Является годом выпуска мыши;
* поле brand. Является брендом мыши;
* поле cost. Является ценой мыши;
* поле mouseSensitivity. Является уровнем чувствительности мыши;
* поле material. Содержит информацию о материале мыши;
* поле cursorFrequency. Содержит уровень частоты курсора;
* поле numberOfSensors. Содержит кол-во датчиков;

# **2.2.2 Файл WiredMouse.txt**

В файле WiredMouse.txt находится основная информация о проводных мышках. Более подробно рассмотрим поля, который хранятся в данном файле:

* поле yearOfIssue. Является годом выпуска мыши;
* поле brand. Является брендом мыши;
* поле cost. Является ценой мыши;
* поле mouseSensitivity. Является уровнем чувствительности мыши;
* поле material. Содержит информацию о материале мыши;
* поле scrollingWay. Содержит информацию о способ прокрутки мыши;
* поле wireLength. Содержит информацию о длине провода мыши;
* поле responseTime. Содержит информацию о времени отклика мыши;

# **2.2.3 Файл LCD.txt**

В файле LCD.txt находится основная информация о LCD-мониторах. Более подробно рассмотрим поля, который хранятся в данном файле:

* поле yearOfIssue. Является годом выпуска мыши;
* поле brand. Является брендом мыши;
* поле cost. Является ценой мыши;
* поле screenResolution. Содержит информацию о разрешение экрана монитора;
* поле numberOfInches. Содержит информацию о кол-ве дюймов монитора;
* поле matrix. Содержит информацию о матрице монитора;
* поле turnOnSpeed. Содержит информацию о времени включения монитора;

# **2.2.4 Файл PDP.txt**

В файле PDP.txt находится основная информация о с. Более подробно рассмотрим поля, который хранятся в данном файле:

* поле yearOfIssue. Является годом выпуска мыши;
* поле brand. Является брендом мыши;
* поле cost. Является ценой мыши;
* поле screenResolution. Содержит информацию о разрешение экрана монитора;
* поле numberOfInches. Содержит информацию о кол-ве дюймов монитора;
* поле plasma. Содержит информацию о установленной плазме на мониторе;
* поле brightnessLevel. Содержит информацию о уровень яркости на мониторе;

# **2.3** **Шаблон программы**

В данной курсовой работе также представлен шаблонный класс двунаправленной очереди(Deque) представляющий из себя класс контейнер, который хранит введенные пользователем объекты.

template <typename T>

struct Node

{

T node;//само значение

Node\* next;//указатель на след элемент

Node\* pred;//указатель на пред элемент

};

В шаблонном классе Deque я реализовал методы:

* Node<T>\* Begin() (голова очереди)
* Node<T>\* End() (хвост очереди)
* void Push(T tmp); (ввод значения)
* int show\_size();(размер очереди)
* void show(int index); (вывод очереди)
* void pop(int index); (удаление очереди с головы или хвоста)
* void pop\_any(T value); (удаление любого элемента очереди)
* void delet(); (очистка очереди)
* vector<T> get();(добавление в вектор значений с итератора)

# **2.4 Класс итератор**

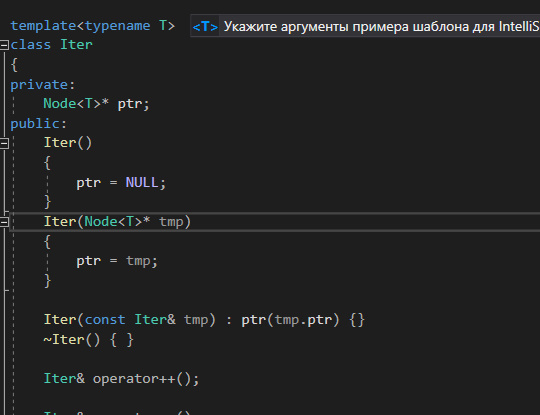
Итератор - это объект, который позволяет перемещаться (итерироваться) по элементам некоторой последовательности.  
Использование таких объектов классифицируется как паттерн программирования Итератор.

Последовательность может быть как готовым набором объектов в памяти, так и состоять объектов которые создаются "на лету" при перемещении итератора (например читаются из файла).

В отличие от разнообразных последовательностей элементов (массивы, списки, файлы), итераторы имеют одинаковый интерфейс: получение текущего элемента, перемещение к следующему. Это позволяет писать более общие алгоритмы, которые работают с любыми итераторами, поддерживающими этот минимальный набор функций.

Также в своей курсовой работе я создал свой собственный класс итератора в котором я реализовал перегрузки для более удобной работы с контейнером :

* Iter& operator++();
* Iter& operator--();
* Iter& operator=(const Node<T>& tmp);
* T& operator\*();
* Node<T>\* operator & ();
* bool operator == (const Node<T>\* tmp);
* bool operator != (const Node<T>\* tmp);



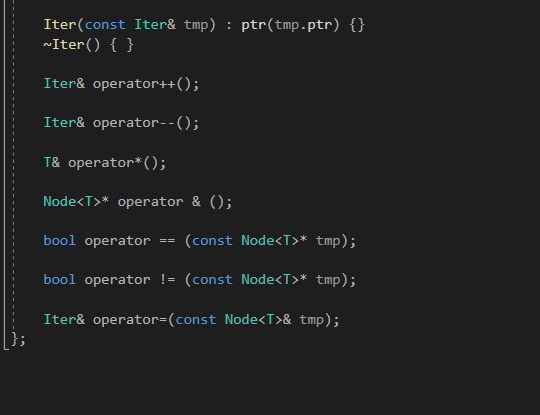


Рисунок 2.3.1 – реализация класса итератора Iter

# **2.5 Cвой класс String**

**string** — класс с методами и переменными для организации работы со [строками](https://ru.wikipedia.org/wiki/%D0%A1%D1%82%D1%80%D0%BE%D0%BA%D0%B0_(%D1%82%D0%B8%D0%BF_%D0%B4%D0%B0%D0%BD%D0%BD%D1%8B%D1%85)) в языке программирования [C++](https://ru.wikipedia.org/wiki/C%2B%2B). Он включён в [стандартную библиотеку C++](https://ru.wikipedia.org/wiki/%D0%A1%D1%82%D0%B0%D0%BD%D0%B4%D0%B0%D1%80%D1%82%D0%BD%D0%B0%D1%8F_%D0%B1%D0%B8%D0%B1%D0%BB%D0%B8%D0%BE%D1%82%D0%B5%D0%BA%D0%B0_C%2B%2B). Название образовано от имени строчного типа данных ([англ.](https://ru.wikipedia.org/wiki/%D0%90%D0%BD%D0%B3%D0%BB%D0%B8%D0%B9%D1%81%D0%BA%D0%B8%D0%B9_%D1%8F%D0%B7%D1%8B%D0%BA) *string*; с [англ.](https://ru.wikipedia.org/wiki/%D0%90%D0%BD%D0%B3%D0%BB%D0%B8%D0%B9%D1%81%D0%BA%D0%B8%D0%B9_%D1%8F%D0%B7%D1%8B%D0%BA) — «строка»).

В данном классе я реализовал библиотечный класс string, в нем реализованны:

friend ostream& operator<<(ostream& on, String& ss);

friend istream& operator>>(istream& in, String& ss);

String& operator=(const String& other);

String& operator+= (String const& other);

String operator()(int index1, int length);

String operator+(const String& other);

bool operator==(const String& tmp);

bool operator == (const char\* tmp);

bool operator!=(const String& other);

bool operator<(const String& other);

bool operator>(const String& other);

char& operator[](int index);

void operator = (char\* str);

void operator = (const char str[]);

const char\* c\_str()

{

return this->str;

}

int slength()

{

return this->length;

# **3 РАЗРАБОТКА ПРОГРАММНЫХ МОДУЛЕЙ**

# **3.1 Алгоритмы по шагам**

Алгоритм метода sort()

template<class T>

void Algorithm<T>::sort(Deque<T>& a)

1. Начало
2. Входные данные
   1. Node<T> \* tmp – указатель на структуру контейнера.
3. Iter<T> it\_beg – итератор на начало контейнера.
4. Iter<T> it\_next – итератор на след. элемент контейнера.
5. Выходные данные
   1. Node<T> \* tmp –отсортированный контейнер.
6. Цикл while c условием(it\_beg != NULL)
   1. it\_next = it\_beg – присваеваем значению итератора на начало контейнера значение итератора на след. элемент контейнера.
   2. ++ it\_next переносим указатель на след. элемент на +1.
7. Условие if((\*it\_beg) > (\*it\_next)) – если значение it\_beg > it\_next , то переходим на 8 шаг.
8. tmp->node = \*it\_beg;

\*it\_beg = \*it\_next;

\*it\_next = tmp->node;

1. ++ it\_next переносим указатель на след. элемент на +1.
2. ++it\_beg переносим указатель на след. элемент на +1.
3. Конец цикла while
4. Конец

Алгоритм поиска search()

template<class T>

Deque<T>& Algorithm<T>::search(Node<T>\* beg, T \_obj)

1. Начало
2. Входные данные
   1. Node<T> \* tmp – указатель на структуру контейнера.
   2. T \_obj – объект контейнера.
3. Node<T> \*\_rab = beg – инициализируем структуру контейнера.
4. Deque<T>temp – контейнер.
5. Выходные данные
   1. Deque<T> temp –найденное значение в контейнере.
6. Цикл while c условием(rab!= NULL)
7. Условие if(rab->node == \_obj) – если значение rab->node == \_obj, то переходим на 8 шаг.
8. сout << rab->node;

temp.Push(rab->node);

1. rab = rab->next; - переносим указатель на след. значение.
2. Возвращаем значение temp.
3. Конец цикла while
4. Конец

# **3.2 Схемы алгоритмов**

Схема алгоритма метода sort()

template<class T>

void Algorithm<T>::sort(Deque<T>& a)

приведена в приложении Б – метод поиск объекта.

Схема алгоритма метода search()

template<class T>

Deque<T>& Algorithm<T>::search(Node<T>\* beg, T \_obj)

приведена в приложении В – метод сортировки шаблона.

# **4 ТЕСТИРОВАНИЕ**

При запуске программы открывается окно

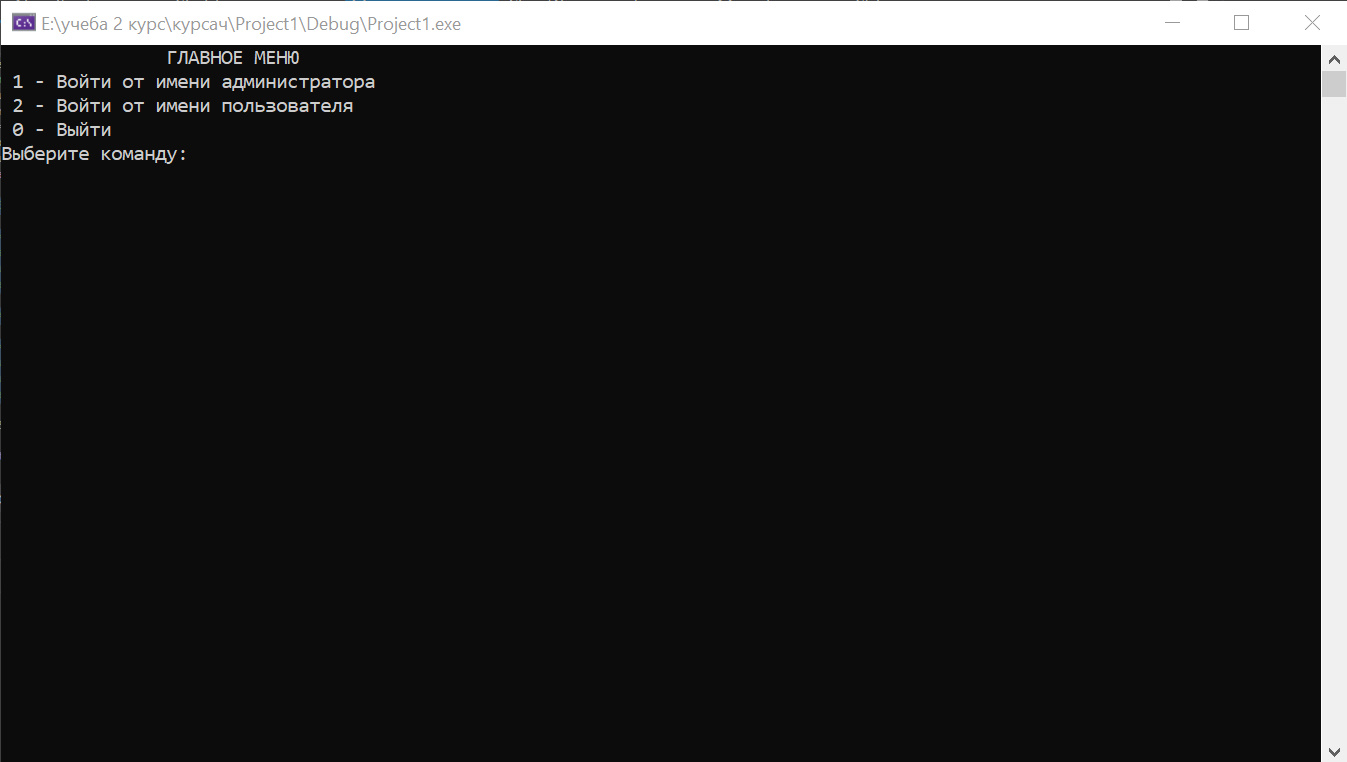


Рисунок 4.1 – Главное меню

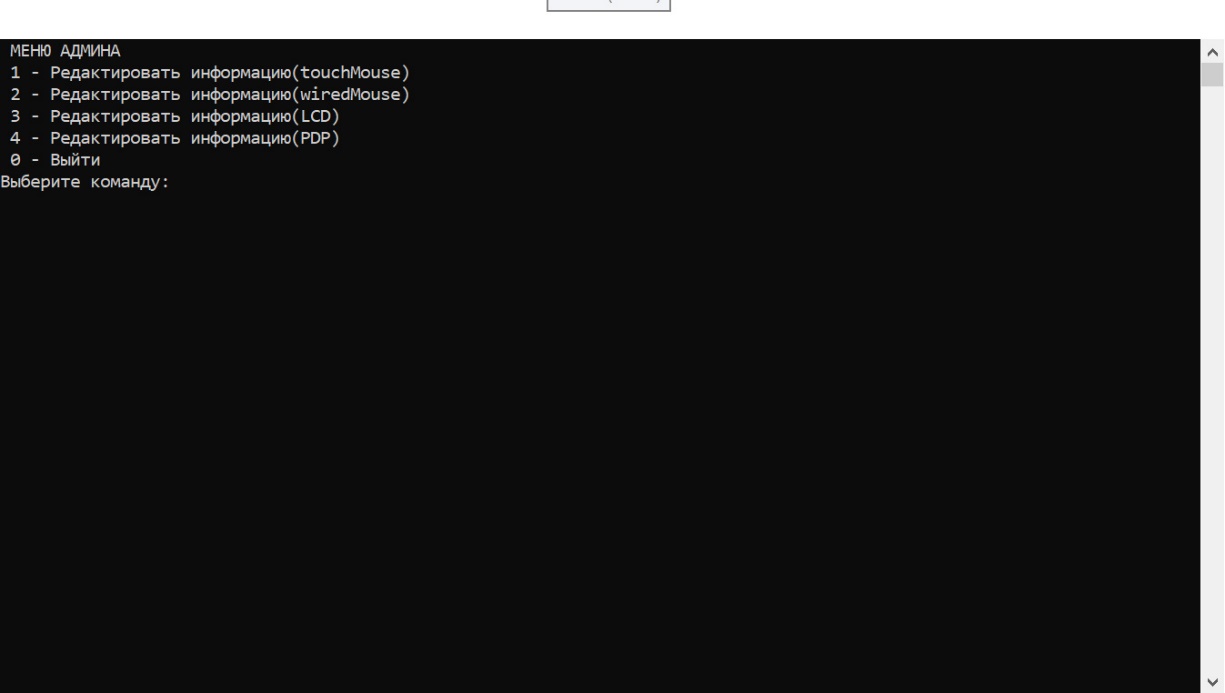


Рисунок 4.2 – Меню администратора

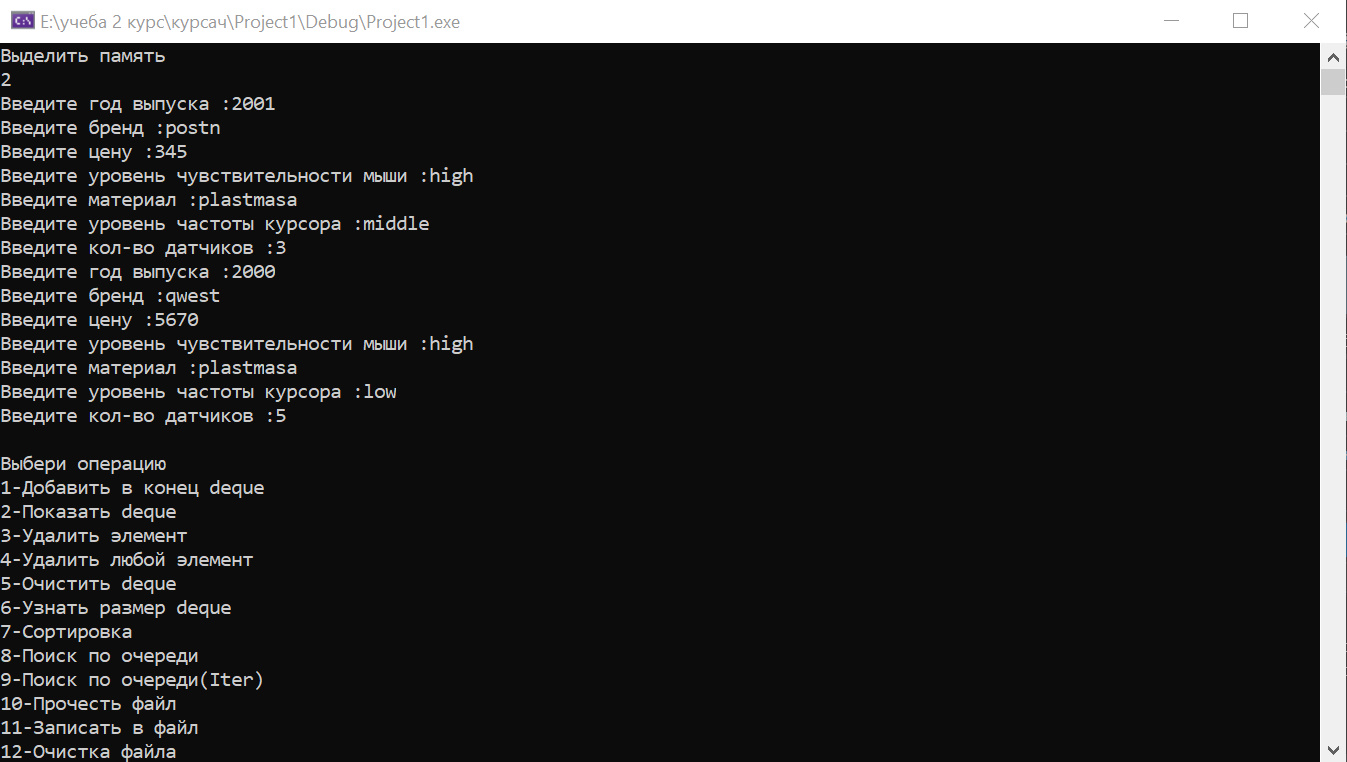


Рисунок 4.3 –Выбор операций для работы с объектами

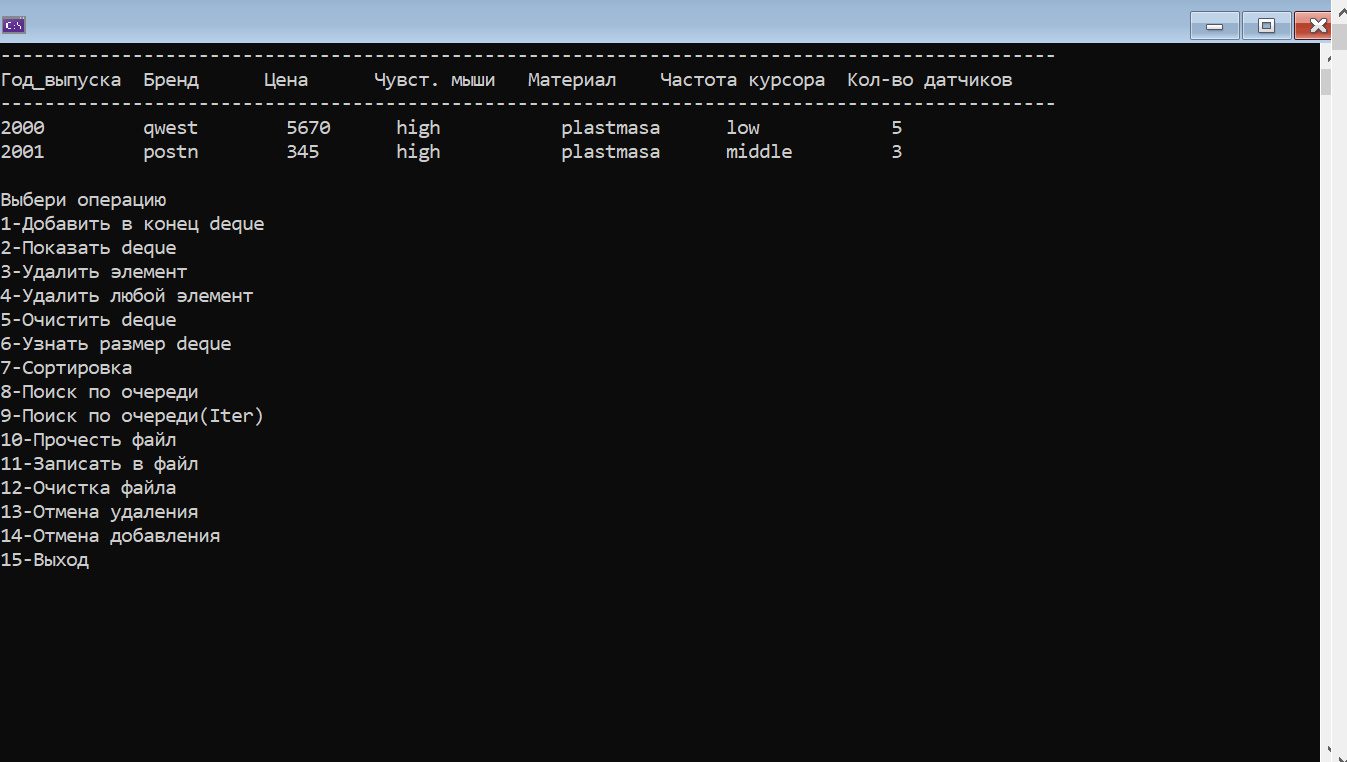


Рисунок 4.4 –Просмотр контейнера

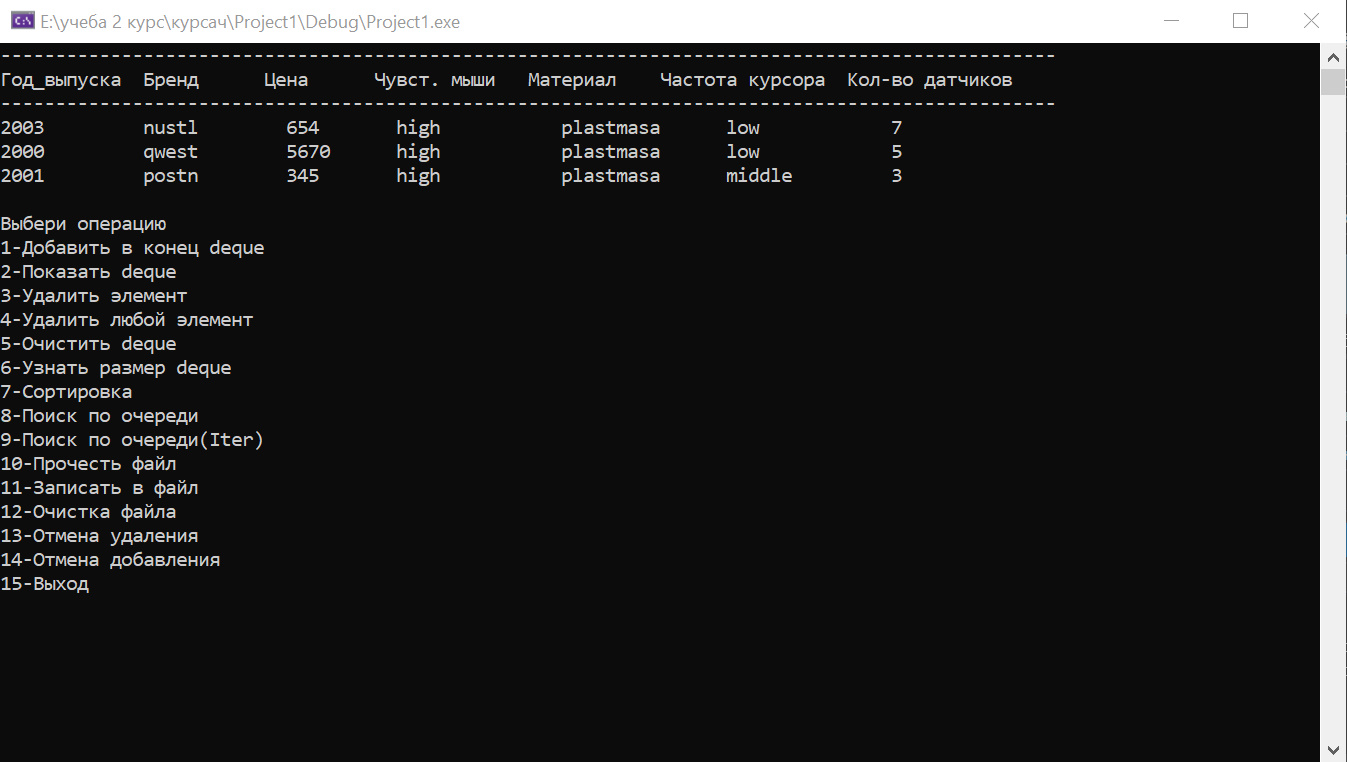


Рисунок 4.5 –Добавление объекта в контейнер

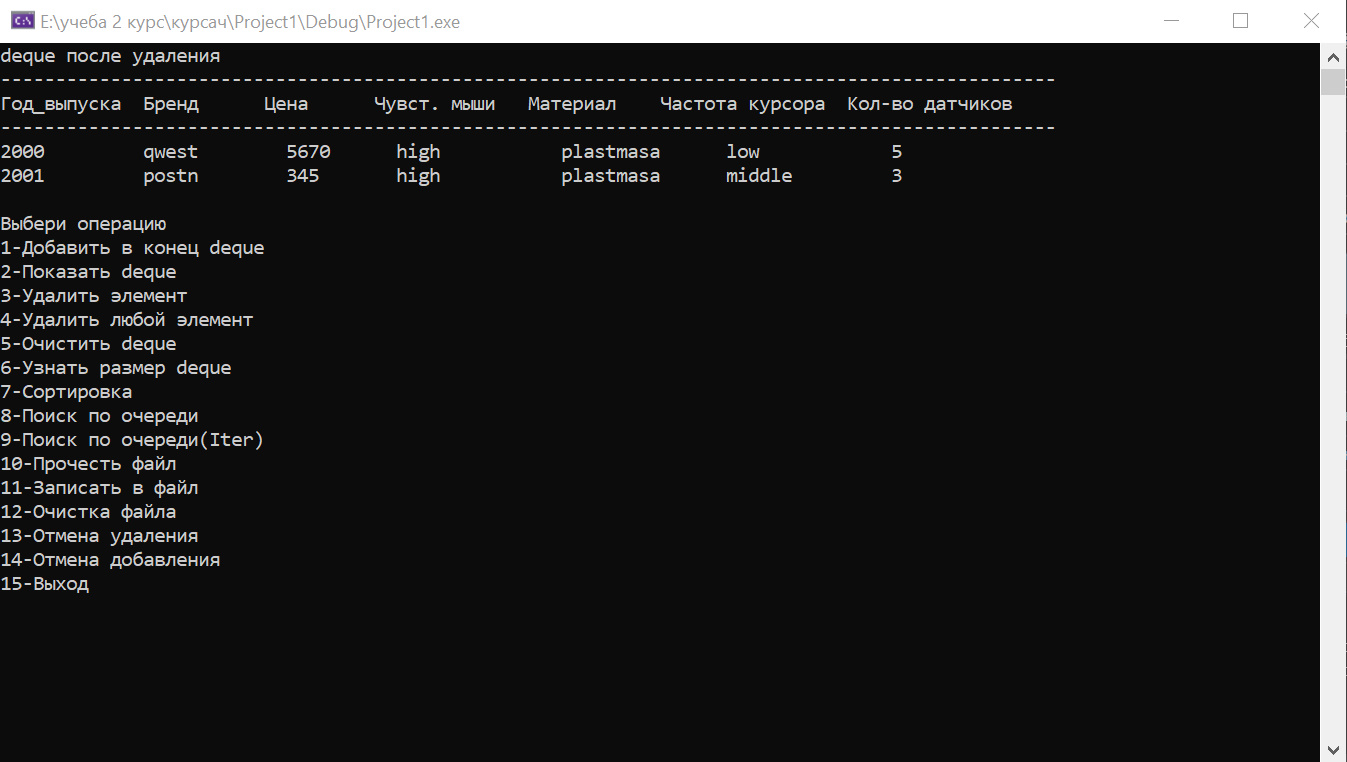


Рисунок 4.6 –Удаление объекта из контейнера

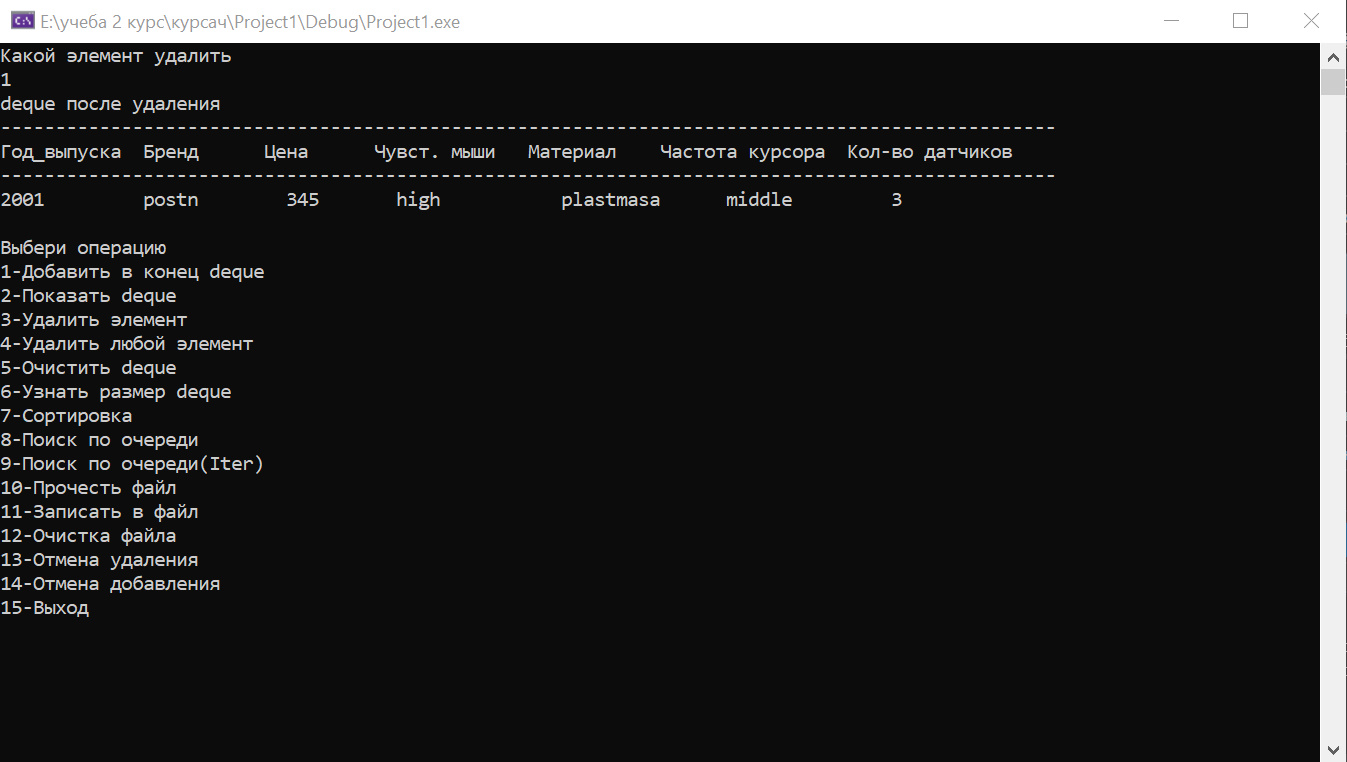


Рисунок 4.7 –Удаление любого объекта из контейнера

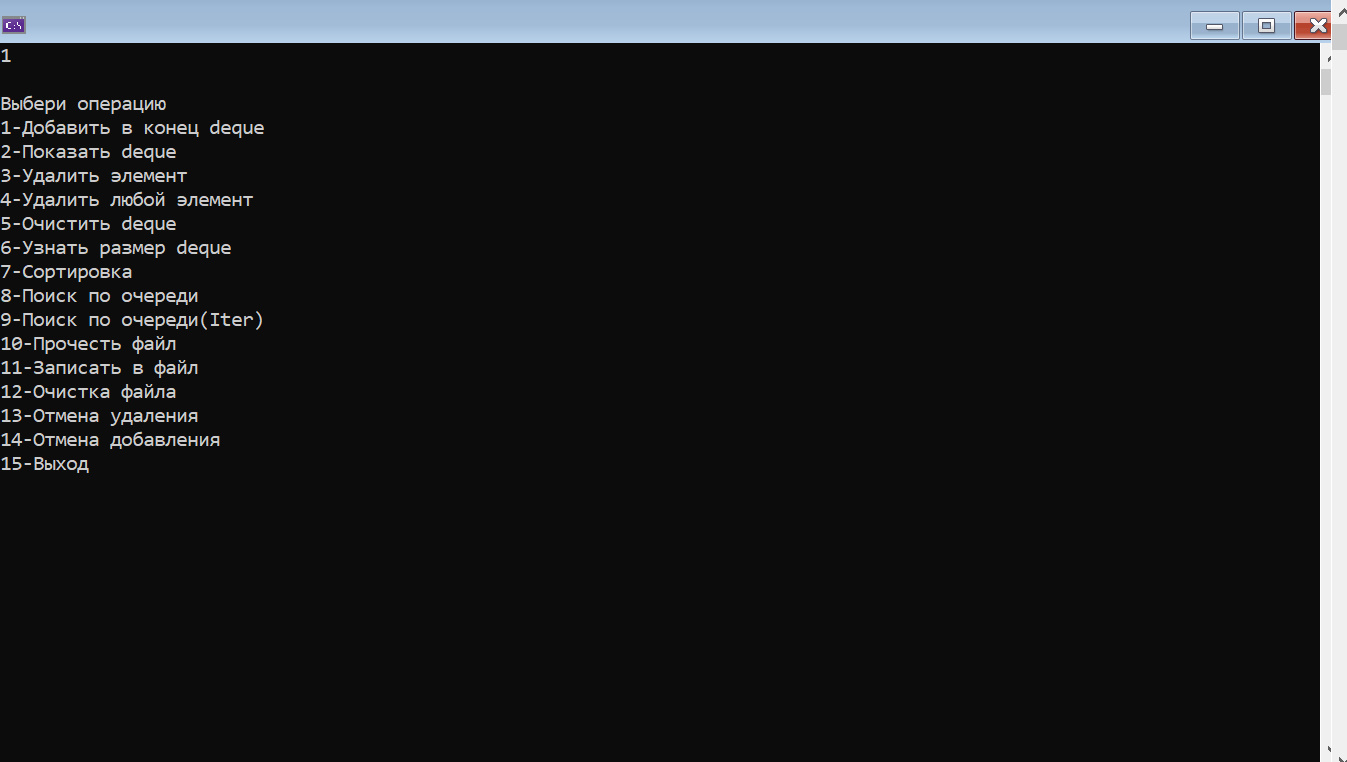


Рисунок 4.8 –Размер контейнера

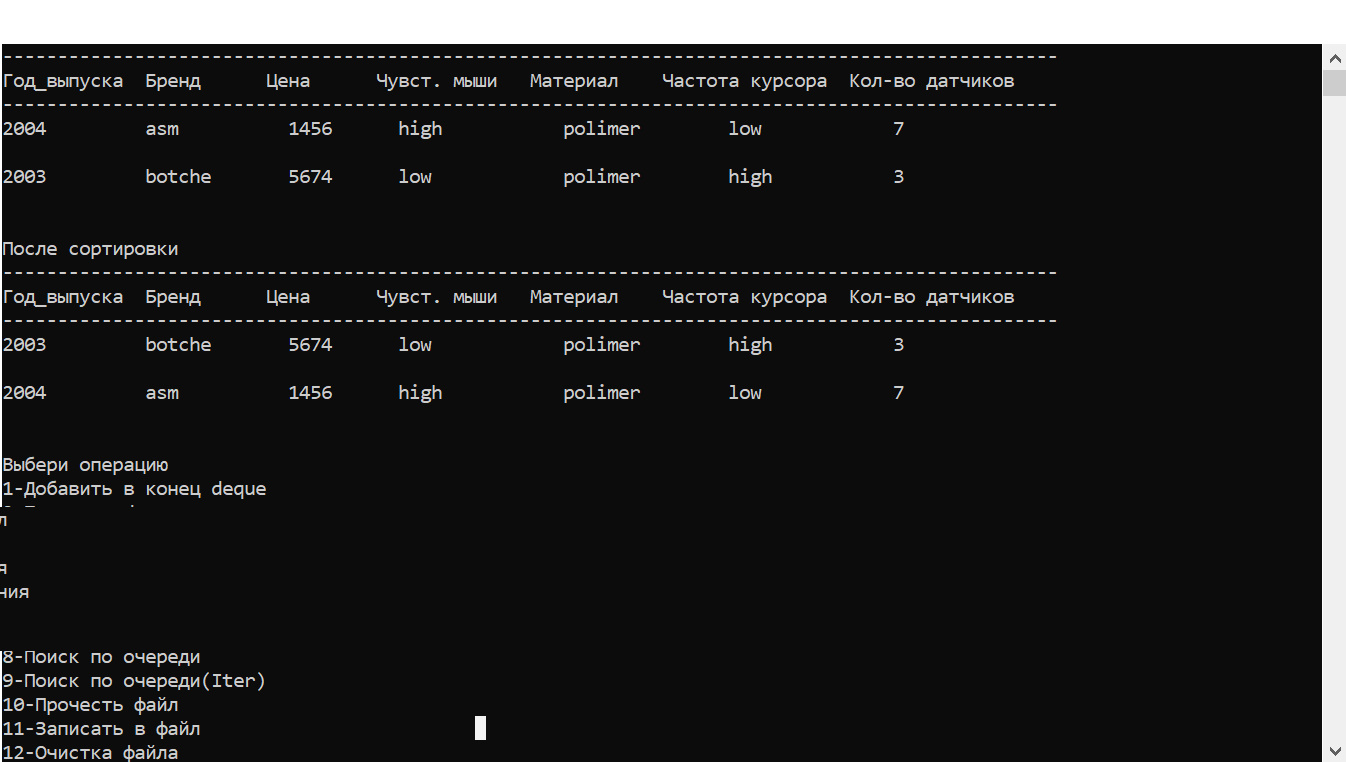


Рисунок 4.9 –Сортировка контейнера

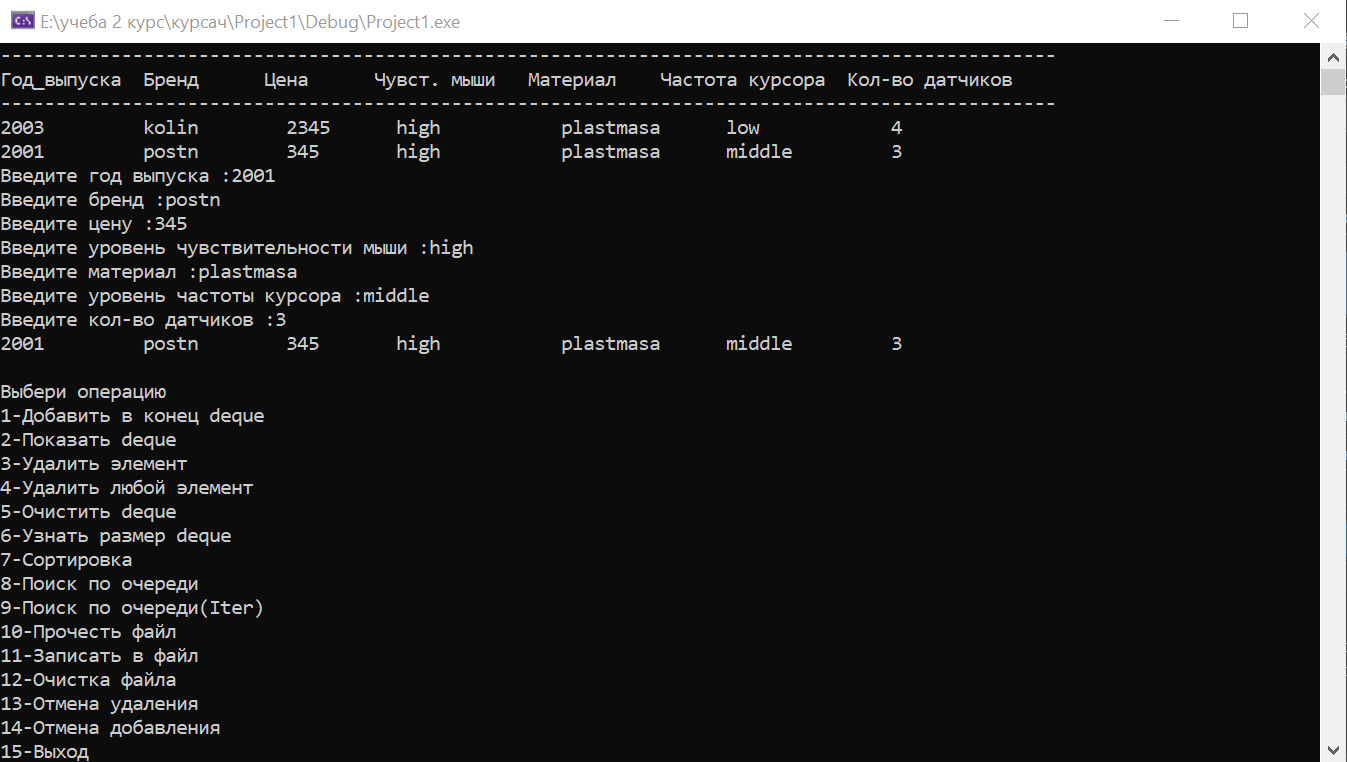


Рисунок 4.10 –Поиск по контейнеру

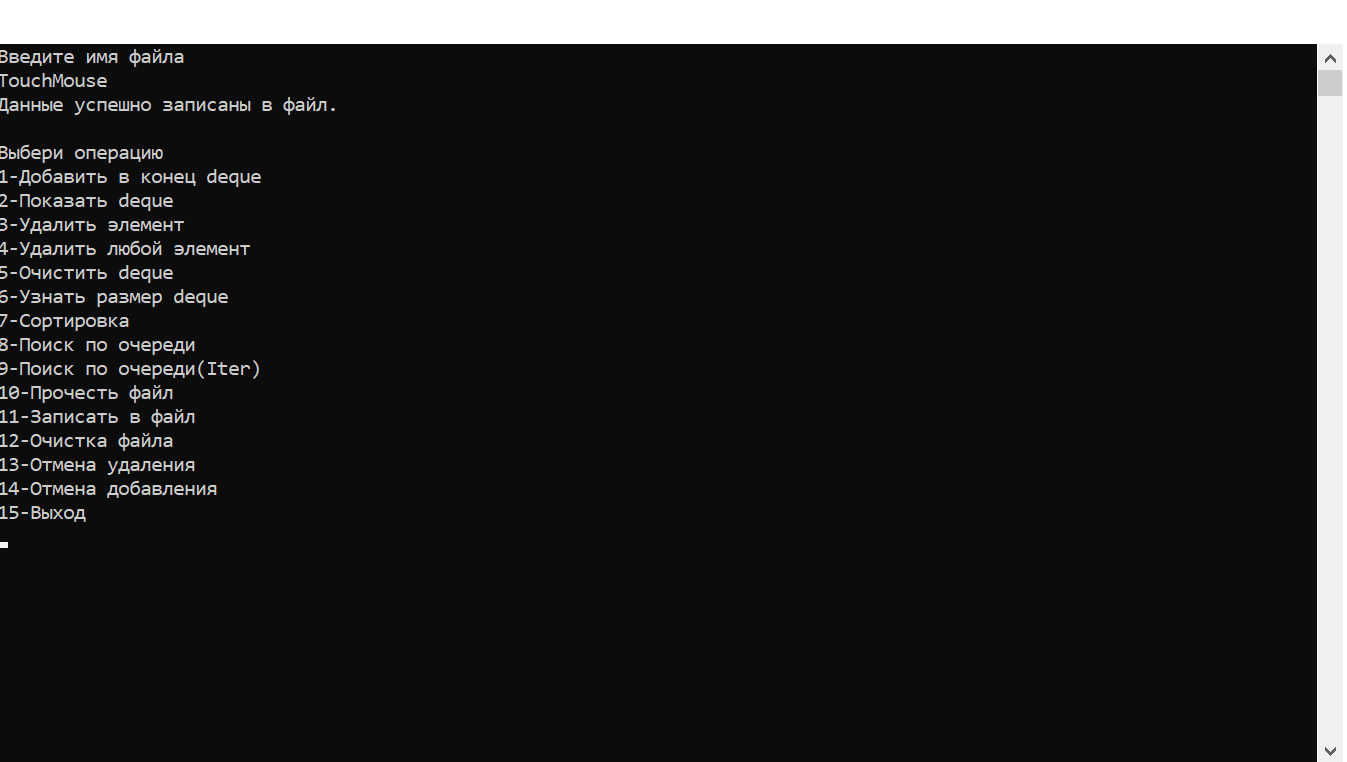


Рисунок 4.11 –Поиск по контейнеру при помощи итератора

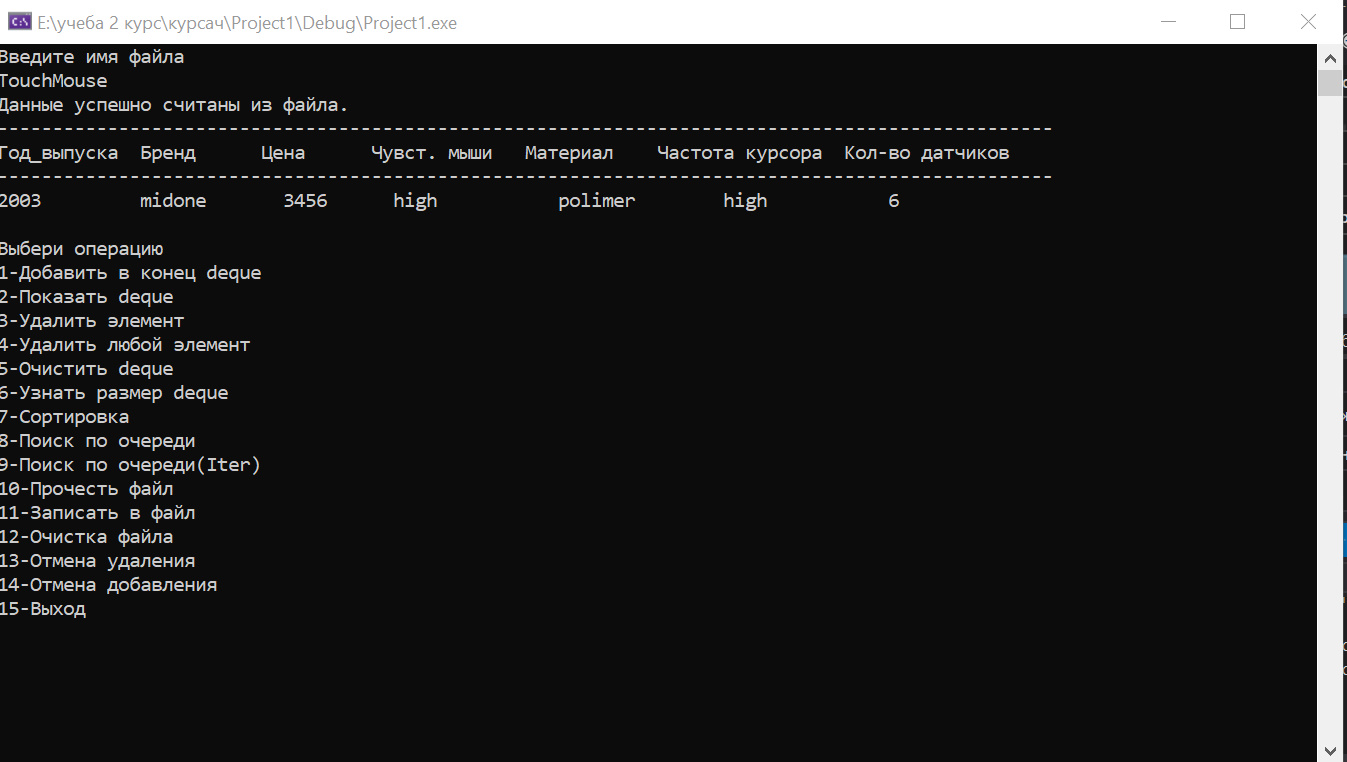


Рисунок 4.12 –Прочесть файл

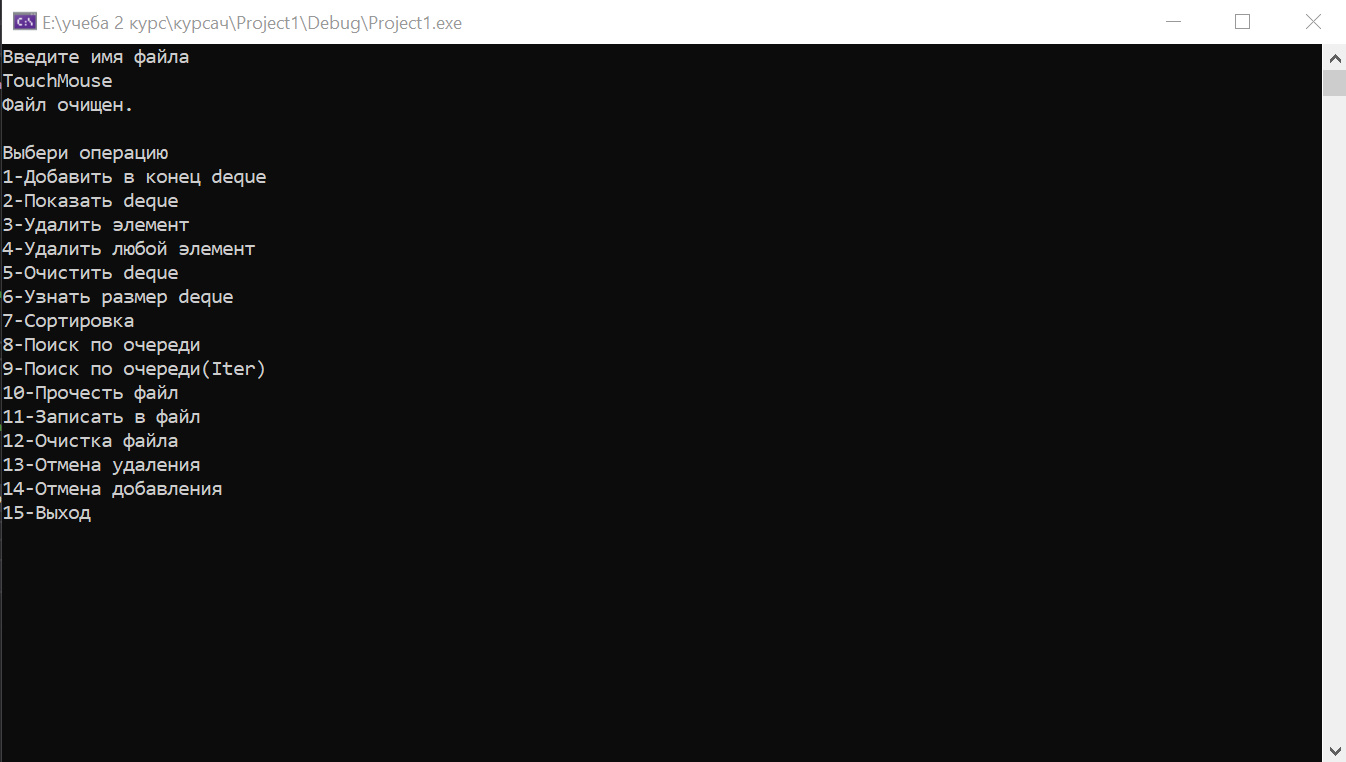


Рисунок 4.13 –Записать в файл

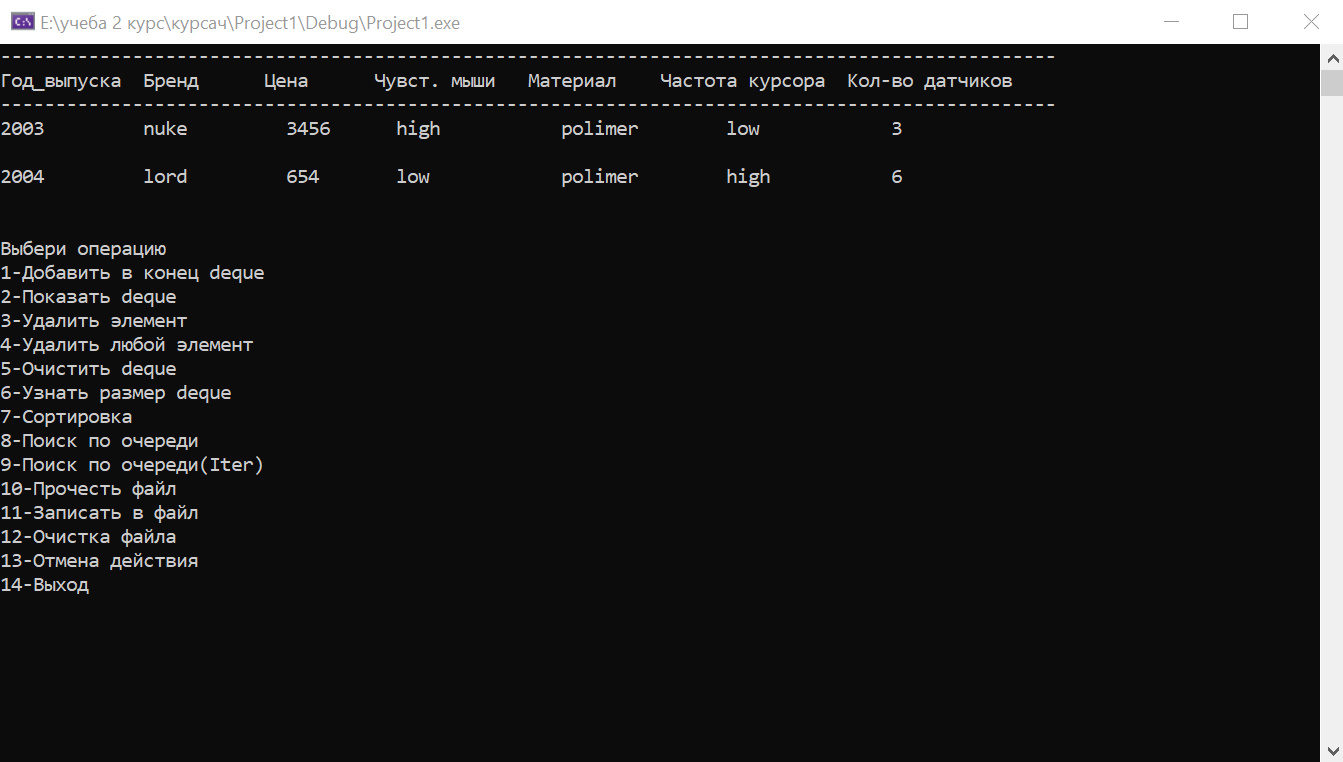


Рисунок 4.14.1 –Контейнер до отмены последнего действия

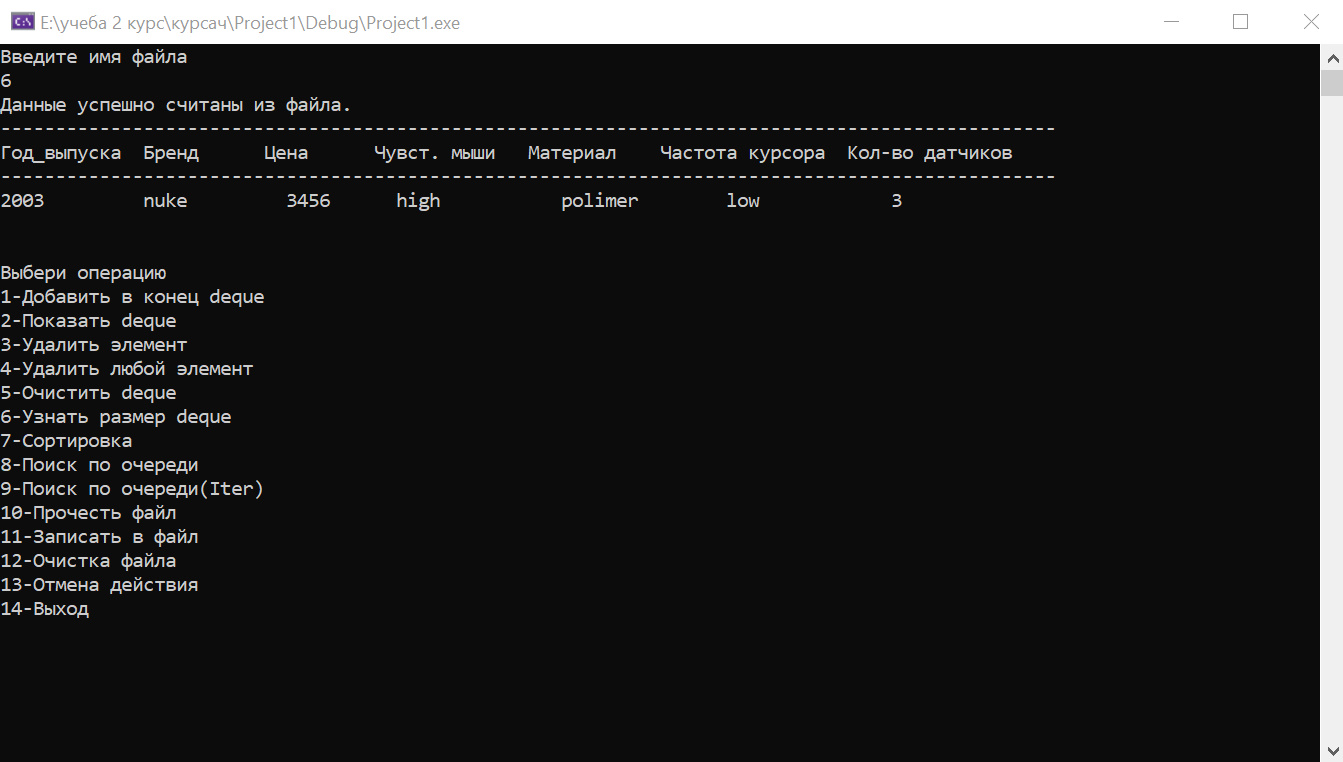


Рисунок 4.14.1 –Контейнер после отмены последнего действия

# 

# **5. СПИСОК ЛИТЕРАТУРЫ**

[1].Язык программирования С++ [Электронный ресурс]. – Режим доступа: <https://ru.wikipedia.org/wiki>/C++. – Дата доступа 19.9.2019.

[2].Языки программирования [Электронный ресурс]. – Режим доступа: <http://sd-company.su/article/help_computers/programming_language> – Дата доступа 5.10.2019.

[3].Технологии программирования [Электронный ресурс]. – Режим доступа: [http://info-master.su/programming/profi/programming-technologies.php – Дата доступа 7.10.2019](http://info-master.su/programming/profi/programming-technologies.php%20–%20Дата%20доступа%207.10.2019).

[4].Структура программного обеспечения ПК [Электронный ресурс]. – Режим доступа: [https://studopedia.ru/4\_4312\_tema--struktura-programmnogo-obespecheniya-pk.html – Дата доступа 20.10.2019](https://studopedia.ru/4_4312_tema--struktura-programmnogo-obespecheniya-pk.html%20–%20Дата%20доступа%2020.10.2019).

[5].Руководство по разработке на компонентах [Электронный ресурс]. – Режим доступа: https://habr.com/ru/company/ruvds/blog/461661/ – Дата доступа 11.10.2019.

[6].Теоретические основы алгоритмизации и программирования [Электронный ресурс]. – Режим доступа: <https://studfile.net/preview/4693947/> – Дата доступа 19.10.2019.

[7].Луцик, Ю. А. Объектно-ориентированное программирование на языке С++: учеб. пособие по курсу «Объектно-ориентированное программирование» для студ. спец. «Вычислительные машины, системы и сети» всех форм обуч. / Ю. А. Луцик, А. М. Ковальчук, И. В. Лукьянова. – Минск: БГУИР, 2003. – 203 с.:ил.

# **ЗАКЛЮЧЕНИЕ**

В результате выполнения курсового проекта была разработано программное средство по управлению сведениями об игроках футбольной команды. Во время реализации данного проекта были использован язык программирования C++. При разработке программы стало понятно, что классы и объекты в С++ являются основными концепциями объектно-ориентированного программирования — ООП. А объектно-ориентированное программирование — расширение структурного программирования, в котором основными концепциями являются понятия классов и объектов. Стало ясно что основное отличие языка программирования С++ от С состоит в том, что в С нет классов, а следовательно язык С не поддерживает ООП, в отличие от С++.

В программе удалось реализовать удобный пользовательский интерфейс, хороший дизайн. Программа проста в использовании и имеет большой функционал, который помогает при работе с сведениями об игроках футбольной команды.

**Приложение А**

*(обязательное)*

Диаграмма классов

**Приложение Б**

*(обязательное)*

Схема алгоритма метода sort()

template<class T>

void Algorithm<T>::sort(Deque<T>& a)

**Приложение В**

*(обязательное)*

Схема алгоритма метода search()

template<class T>

Deque<T>& Algorithm<T>::search(Node<T>\* beg, T \_obj)

**Приложение Г**

(*обязательное*)

Код программы

**Приложение Д**

*(обязательное)*

Ведомость документов