«Московский авиационный институт»   
(Национальный исследовательский университет)

Отчет по учебно-исследовательской работе студентов Тема: сетевая структура предприятий

Выполнила бригада:

Двоеглазова М.В.

Понкращенков Д.Б.

Проверил:

Барчев Н. Б.

**Содержание**

1. Задание. ………………………..………………………………………………………………………………….……...3
2. Псевдокод. ………………………….…….…………………………………………………………………….….…...4
3. Сведения о программной реализации.  …………………….……………………………………….....9
4. Инструкция пользователя. …………………………….……………………………………………………...15
5. Тестирование. .………………………………………………………………….…………………………………...20
6. Листинг. ………………………………………………….……………………………………………………………...25
7. **Задание**

Написать программу сетевой структуры предприятий с графическим интерфейсом, в которой будет возможность создавать такие сущности как "Объект" (предприятия).

Объектам присуще такие поля как:

* Имя объекта (Промышленности)
* Список ресурсов, производимых конкретным объектом.
* Список поставок, поставляемых в этот объект.

Необходимо выполнить:

* Реализовать возможность добавления новых промышленностей (Объектов), и их удаления.
* Реализовать возможность добавления ресурсов в конкретную промышленность, а также их удаление.
* Реализовать возможность добавления поставок ресурса в конкретную промышленность, а также их удаление.
* Выводить всю существующую структуру предприятий в текстовый файл со всеми существующими объектами и поставками.

1. **Псевдокод**

**Функция main:**

* Пытаемся открыть файл, для чтения данных и занесения в лист.
* Выводим меню:

1. Добавить промышленность
2. Добавить ресурс
3. Добавить поставку
4. Удалить промышленность
5. Удалить ресурс
6. Удалить поставку
7. Вывести таблицу поставок
8. Сохранить всё в файл

(ESC) Завершить программу

* Если выбран 1 п. меню:
* Считываем имя промышленности.
* Вызываем функцию добавления объектов в лист.
* Выводим существующий список.
* Предлагаем вернуться в меню или добавить еще промышленность.
* Если выбран 2 п. меню:
* Спрашиваем, в какую промышленность необходимо добавить ресурс.
* Проверяем, есть-ли объект под таким номером, если нет:
  + Предложить выйти в меню.
  + Ввести номер ещё раз.
* Если объект под таким номером существует, запрашиваем имя ресурса.
* Вызываем функцию добавления ресурсов в лист.
* Выводим обновлённый список.
* Предлагаем выйти в меню или добавить еще ресурс.
* Если выбран 3 п. меню:
* Спрашиваем номер промышленности, откуда пойдёт поставка.
* Проверяем, есть-ли объект под таким номером, если нет:
  + Предложить выйти в меню.
  + Ввести номер ещё раз.
* Спрашиваем номер ресурса.
* Проверяем, есть ли ресурс под таким номером, если нет, спросить ещё раз.
* Спрашиваем номер объекта, куда пойдёт поставка.
* Проверяем, есть ли объект под таким номером, если нет, спросить ещё раз.
* Вызываем функцию добавления поставок в лист.
* Предлагаем выйти в меню или добавить еще поставку.
* Если выбран 4 п. меню:
  + Спрашиваем номер промышленности и проверяем его.
  + Если номер верный, запрашиваем подтверждение на удаление
  + Вызываем функцию удаления из листа объектов.
  + Выводим обновленный лист.
  + Предлагаем выйти в меню или удалить ещё один объект.
* Если выбран 5 п. меню:
  + Спрашиваем номер промышленности и проверяем его.
  + Спрашиваем номер ресурса и проверяем его.
  + Если номер верный, запрашиваем подтверждение на удаление
  + Вызываем функцию удаления из листа ресурсов.
  + Выводим обновленный лист.
  + Предлагаем выйти в меню или удалить ещё один объект.
* Если выбран 6 п. меню:
  + Спрашиваем номер промышленности и проверяем его.
  + Спрашиваем номер ресурса и проверяем его.
  + Спрашиваем номер поставки и проверяем его
  + Если номер верный, запрашиваем подтверждение на удаление
  + Вызываем функцию удаления из листа поставок.
  + Выводим обновленный лист.
  + Предлагаем выйти в меню или удалить ещё один объект.
* Если выбран 7 п. меню:
  + Вызываем функцию вывода всех промышленностей/ресурсов/поставок.
* Если выбран 8 п. меню:
  + Вызываем функцию записи в файл.
* Если выбран 8 п. меню:
  + Завершаем главный цикл меню.

**Файл List.h:**

Поскольку основные функции по типу add/del для обработки ресурсов и промышленностей заключают в себе принцип «Пролистал список до нужного номера, для получения необходимых адресов объектов и вызвал функцию-шаблон», то они не требуют псевдокода для представления и понимания особого принципа работы.

В случае обработки поставок в функциях Add\_delivery/dell\_Deliv производится всё таже операция по получению нужных адресов промышленности, откуда идёт поставка, поставляемого ресурса и промышленности, куда направленна поставка, после чего:

* Для запоминания, куда поставляется ресурс – создать список, хранящий адреса объектов.
* Для запоминания, откуда и какой приходит ресурс в объект – создать список, хранящий адреса поставляемых ресурсов и объектов, их хранящих.

**Функция ShowAll:** //Функция вывода данных в консоль.

* Проверить, есть-ли хоть один объект в списке
  + Если нет – вывести сообщение, что список пуст.
  + Пока не достигнут конец списка промышленностей:
    - Вывести номер в списке и имя промышленности.
    - Если есть ресурсы и пока не достигнут конец списка ресурсов:
      * Вывести номер в списке и имя ресурса.
      * Если есть поставки и пока не достигнут конец списка поставок:
        + Вывести номер в списке и имя объекта, куда идёт поставка.
        + Перейти к следующей поставке.
      * Перейти к следующему ресурсу.
    - Если есть поставки эту промышленность:
      * Вывести имя поставляемого ресурса и объекта, его поставляющего.
      * Перейти к следующей приходящей поставке.
    - Перейти к следующей промышленности.

**Функция OpenFile:** //Функция чтения данных из файла.

* Открыть файл «Delivery.txt» для чтения.
* Считывание первых 2-х строчек файла.
* Пока не достигнут конец файла: Цикл № 1
* Переводим маркер с нумерации промышленности.
* Считываем имя промышленности целиком (1)
* Добавляем имя промышленности в список
* Определяем, есть ли ресурсы у промышленности
* Если встречено "Ресурсы:" – это список ресурсов
  + - Считываем для перевода маркера с нумерации ресурса
    - Цикл считывания ресурсов, пока не дойдём до конца списка ресурсов Цикл № 2
      * Считываем имя ресурса целиком
      * Добавляем имя ресурса в список
      * Считаем текущий номер ресурса в списке ресурсов для организации поставок (2)
      * Фиксируем, есть ли поставка / следующий ресурс
      * Если обнаружено слово "Осуществляются" – это список поставок
      * Считываем для перевода маркера с нумерации поставок
      * Пока это нумерованный список поставок "|1."   
        определяем его по символу «|» перед нумерацией. Цикл № 3
        1. Считываем имя поставки (т.е. промышленности, в которую идёт поставка) (3)
        2. Сохраняем индекс промышленности (1), номер ресурса (3) и имя поставки (3) во временную структуру - буфер, чтоб потом оформить поставки. (!!!)
        3. Считываем следующее слово и смотрим, что дальше.   
           (Возвращаемся в начало тела цикла №3)
* Если обнаружена нумерация – это следующий по списку ресурс.
  + Возвращаемся в начало тела цикла № 2 и начинаем считывать имя ресурса.
* Если обнаружено "----------" – это конец блока записей для текущей промышленности.
  + Возвращаемся в начало тела цикла № 1 и начинаем считывать имя промышленности.
* Если обнаружено "Поставки" – это список поставок, приходящих в эту промышленность.
  + Считываем строки целиком, пока не достигнем конца блока.
  + Возвращаемся в начало тела цикла № 1 и начинаем считывать имя промышленности.
* Если встречено "Ресурсов нет" – значит ресурсов, производимых промышленностью нет.
  + Считываем следующую запись.
* Если следующая запись - "Поставки" – это список поставок, приходящих в эту промышленность.
  + Считываем строки целиком, пока не достигнем конца блока.
  + Возвращаемся в начало тела цикла № 1 и начинаем считывать имя промышленности.
* Если нет, то это конец блока, считываем второй разделитель на блоки.

\*Достигаем конца файла\*

* Вызываем функцию добавления поставок и подаём в неё индексы из временной   
  структуры-буфера (!!!)

1. **Сведения о программной реализации:**

Версия системы программирования:   
Microsoft Visual Studio Community 2022 Version 143  
  
Требуемые для работы ресурсы вычислительной системы:

|  |  |
| --- | --- |
| Поддерживаемые операционные системы: | * Windows 7 и выше |
| Процессор | * Процессор с тактовой частотой не ниже 1,8 ГГц. |
| Оперативная память | * 48 кб ОЗУ, рекомендуется 164 кб ОЗУ |

**Описание входных и выходных данных:**

Входные данные для работы программы не обязательны, но в их качестве может быть файл, полученный в результате работы данной программы в прошлом, расположенный в одной папке с программой:

Delivery.txt

Выходными данными будет текстовый файл, автоматический созданный в репозитории программы, хранящий в себе всю структуру данных о поставках, добавленных пользователем.

**Особые указания:**

* В названии ресурсов/промышленностей крайне не рекомендуется использовать цифры и символ «|».

**Программные единицы:**

Функции класса List:  
---------------------------------------------------------------------------------------------------------------------

Функция *Add\_Obj (string* *newName)*

Назначение: Добавление промышленностей.

Тип: *void*;

Входные параметры: «*newName*» - Имя новой промышленности. Тип: *string*;

Возвращаемое значение отсутствует.

---------------------------------------------------------------------------------------------------------------------

Функция *Add\_ObjRes (**string* *newNameRes, int* *countInd)*

Назначение: Добавление ресурса в объект.

Тип: *void*;

Входные параметры: «*newNameRes*» - Имя добавляемого ресурса. Тип: *string*;

«*countInd*» - Номер пром., производящий ресурс. Тип: *int*;

Возвращаемое значение отсутствует.

---------------------------------------------------------------------------------------------------------------------

Функция *Add\_delivery (int nomerObj, int inObj, int nomerRes)*

Назначение: Добавление поставок.

Тип: *void*;

Входные параметры: «*nomerObj*» - Номер пром. откуда поставляется ресурс. Тип:*int*;

«*inObj*» - Номер пром. куда поставляется ресурс. Тип: *int*;

«*nomerRes*» - Номер поставляемого ресурса. Тип: *int*;

Возвращаемое значение отсутствует.

---------------------------------------------------------------------------------------------------------------------

Функция *Show ()*

Назначение: Вывод всех промышленностей.

Тип: *void*;

Входные параметры: отсутствуют.

Возвращаемое значение отсутствует.

---------------------------------------------------------------------------------------------------------------------

Функция *Show (int* *nomerObj)*

Назначение: Выводит конкретный объект и принадлежащие ему ресурсы Тип: *void*;

Входные параметры: «*nomerObj*» - Номер объекта, который нужно вывести. Тип:*int*;

Возвращаемое значение отсутствует.

---------------------------------------------------------------------------------------------------------------------Функция *ShowAll ()*

Назначение: Вывод всех объектов/ресурсов/поставок.

Тип: *void*;

Входные параметры: отсутствуют.

Возвращаемое значение отсутствует.

---------------------------------------------------------------------------------------------------------------------

Функция *OpenFile**()*

Назначение: Считывание данных из файла и запись в лист.

Тип: *void*;

Входные параметры: отсутствуют.

Возвращаемое значение отсутствует.

---------------------------------------------------------------------------------------------------------------------

Функция *SaveFile**()*

Назначение: Запись данных из листа в файл.

Тип: *void;*

Входные параметры: отсутствуют;

Возвращаемое значение отсутствует.

---------------------------------------------------------------------------------------------------------------------

Функция *dell\_Obj**(int nomerObj)*

Назначение: Удаление объекта из списка промышленностей.

Тип: *void*;

Входные параметры: «*nomerObj*» - Номер удаляемой промышленности. Тип:*int*;

Возвращаемое значение отсутствует.

---------------------------------------------------------------------------------------------------------------------

Функция *dell\_Res**(int* *nomerObj, int* *nomerRes)*

Назначение: Удаление ресурса из объекта.

Тип: *void*;

Входные параметры: «*nomerObj*» - Номер пром. откуда удаляется ресурс. Тип: *int*;

«*nomerRes*» - Номер удаляемого ресурса. Тип: *int*;

Возвращаемое значение отсутствует.

---------------------------------------------------------------------------------------------------------------------

---------------------------------------------------------------------------------------------------------------------

Функция *dell\_Deliv (int nomerObj, int nomerRes, int nomerDeliv)*

Назначение: Удаление поставок из списка поставок.

Тип: *void*;

Входные параметры: «*nomerObj*» - Номер пром. откуда поставляется ресурс. Тип:*int*;

«*nomerRes*»- Номер пром. куда поставляется ресурс. Тип: *int*;

«*nomerDeliv*» - Номер поставки. Тип: *int*;

Возвращаемое значение отсутствует.

---------------------------------------------------------------------------------------------------------------------

Функция *get\_NomberObj (string str)*

Назначение: Выдает порядковый номер объекта по имени из списка промышленностей.

Тип: *int*;

Входные параметры: «*str*» - Имя промышленности. Тип: *string*;

Возвращаемое значение: «*i*» - Порядковый номер объекта в списке пром. Тип:*int*;

---------------------------------------------------------------------------------------------------------------------

Функция *check\_EndBlok (string str)*

Назначение: определяет, является ли строка – границей блока.

Тип: *bool*;

Входные параметры: «*str*» - Проверяемая строка. Тип: *string*;

Возвращаемое значение: «*true*» - Да, строка – граница блока. Тип: *bool*;

«*false*» - Нет, строка – не граница блока. Тип: *bool*;

---------------------------------------------------------------------------------------------------------------------

Функция *check\_Numbering (string str)*

Назначение: определяет, является ли строка – нумерацией строки.

Тип: *bool*;

Входные параметры: «*str*» - Проверяемая строка. Тип: *string*;

Возвращаемое значение: «*true*» - Да, строка – нумерация. Тип: *bool*;

«*false*» - Нет, строка – нумерация. Тип: *bool*;

---------------------------------------------------------------------------------------------------------------------

---------------------------------------------------------------------------------------------------------------------

Функция *checkInd (int* *nomerObj)*

Назначение: определяет, есть ли промышленность под таким номером.

Тип: *int*;

Входные параметры: «*nomerObj*» - Проверяемый номер. Тип: *int*;

Возвращаемое значение: «*1*» - Да, такой номер есть. Тип: *int*;

«*0*» - Нет, такого номера нет. Тип: *int*;

---------------------------------------------------------------------------------------------------------------------

Функция *checkRes (int nomerObj, int nomerRes)*

Назначение: определяет, есть ли ресурс под таким номером.

Тип: *int*;

Входные параметры: «*nomerObj*» - Номер промышленности. Тип: *int*;

«*nomerRes*» - Проверяемый номер. Тип: *int*;

Возвращаемое значение: «*1*» - Да, такой номер есть. Тип: *int*;

«*0*» - Нет, такого номера нет. Тип: *int*;

---------------------------------------------------------------------------------------------------------------------

Функция *checkDeliv (int nomerObj, int nomerRes*, *int* *nomerDeliv)*

Назначение: определяет, есть ли поставка под таким номером.

Тип: *int*;

Входные параметры: «*nomerObj*» - Номер промышленности. Тип: *int*;

«*nomerRes*» - Номер ресурса. Тип: *int*;

«*nomerDeliv*» - Проверяемый номер. Тип: *int*;

Возвращаемое значение: «*1*» - Да, такой номер есть. Тип: *int*;

«*0*» - Нет, такого номера нет. Тип: *int*;

---------------------------------------------------------------------------------------------------------------------

Функции-шаблоны:

---------------------------------------------------------------------------------------------------------------------

Функция *delList (int nomber, T\*& head, T\*& tail)*

Назначение: Выполнение типовой операции удаления из двухсвязного списка.

Тип: *void*;

Входные параметры: «*nomber*» - Номер удаляемого объекта. Тип:*int*;

«*head*»- Голова списка. Тип: *T\*&*;

«*tail*» - Хвост списка. Тип: *T\*&*;

Возвращаемое значение отсутствует.

---------------------------------------------------------------------------------------------------------------------

Функция *addList (T\*& head, T\*& tail, T\* obj)*

Назначение: Выполнение типовой операции занесения в двухсвязный список.

Тип: *void*;

Входные параметры: «*obj*» - Добавляемая структура. Тип: *T\**;

«*head*»- Голова списка. Тип: *T\*&*;

«*tail*» - Хвост списка. Тип: *T\*&*;

Возвращаемое значение отсутствует.

---------------------------------------------------------------------------------------------------------------------

Функции структуры obj:

---------------------------------------------------------------------------------------------------------------------

Функция *addRes (string newNameRes)*

Назначение: Добавление ресурса в список ресурсов конкретной промышленности.

Тип: *void*;

Входные параметры: «*newNameRes*» - Имя ресурса. Тип: *string*;

Возвращаемое значение отсутствует.

---------------------------------------------------------------------------------------------------------------------

Функция *show ()*

Назначение: Вывод списка ресурсов конкретной промышленности.

Тип: *void*;

Входные параметры: отсутствуют.

Возвращаемое значение отсутствует.

---------------------------------------------------------------------------------------------------------------------

Функция *addDeliv (obj\*&* *fromObj, resources\*& tempRes)*

Назначение: Добавление объекта из которого идет поставка.

Тип: *void*;

Входные параметры: «*fromObj*» - Адрес поставляющей промышленности, Тип: *obj\*&*;

«*tempRes*» - Адрес поставляемого ресурса, Тип: *resources\*&;*

Возвращаемое значение отсутствует.

---------------------------------------------------------------------------------------------------------------------

Функции структуры resources:

---------------------------------------------------------------------------------------------------------------------

Функция *addDelivFrom (obj\*& inObj)*

Назначение: Добавление поставок в лист объекта в который идет поставка.

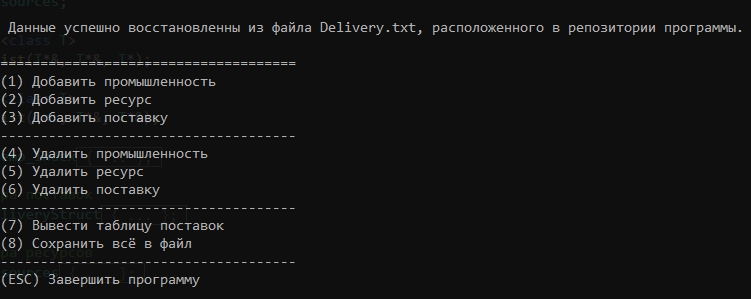
Тип: *void*;

Входные параметры: «*inObj*» - Адрес принимающей промышленности, Тип: *obj\*&*;

Возвращаемое значение отсутствует.

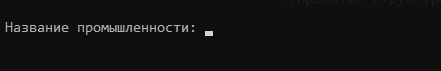
---------------------------------------------------------------------------------------------------------------------

1. **Инструкция пользователя:**

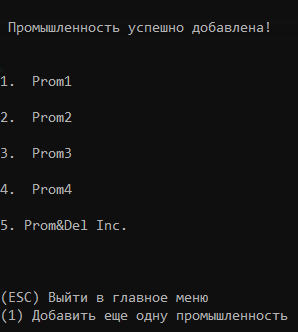
После запуска программы на экране монитора отображается главное меню программы:

В случае, если в директории программы был обнаружен файл Delivery.txt, с него считываются файлы и выводится сообщение, об успешном прочтении.

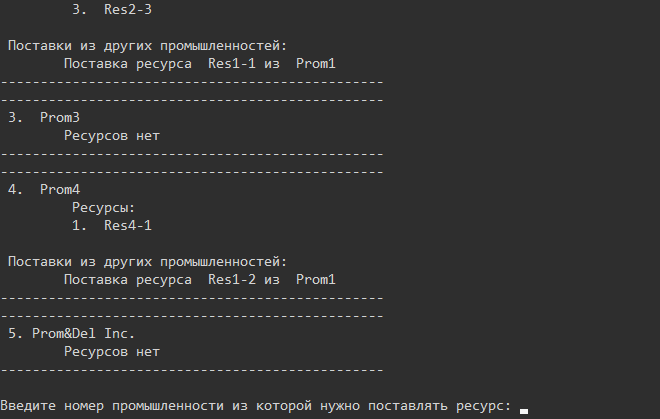
Выбор пунктов осуществляется с помощью ввода нужного номера пункта с клавиатуры.

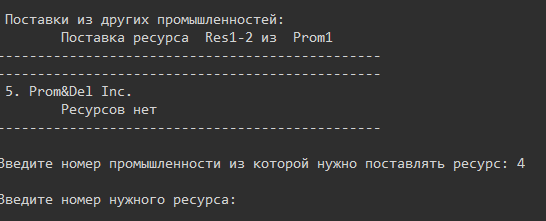
После выбора пользователем, например, первого пункта меню на экране ему представляется возможность ввести название промышленности:

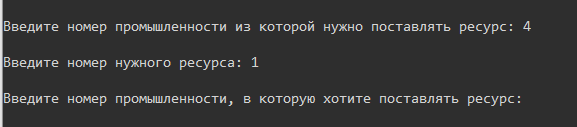
И после его введения и нажатия кнопки enter выведется сообщение об успешном добавлении новой промышленности, а также весь список всех промышленностей.



Далее пользователю представляются возможность выйти в меню по нажатию «esc» или продолжить добавление промышленностей «1».

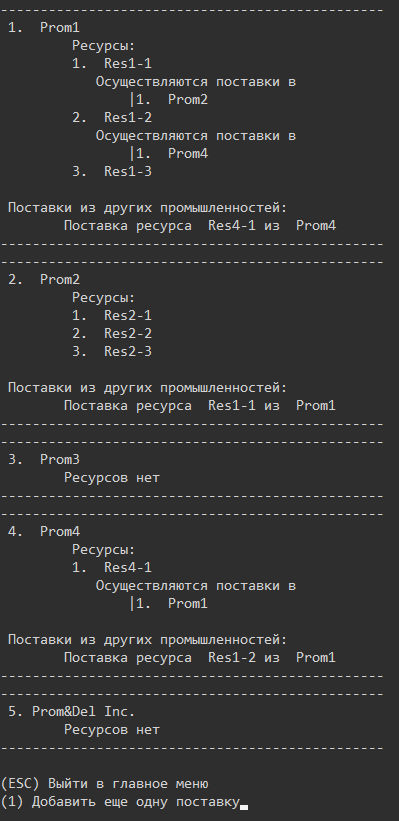
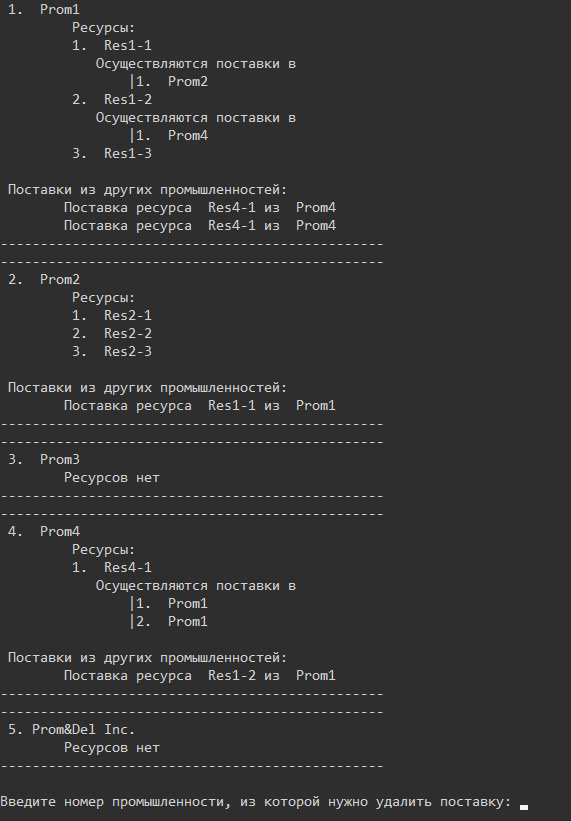
Чтобы добавить, например, поставку, выберем 3 пункт в главном меню:

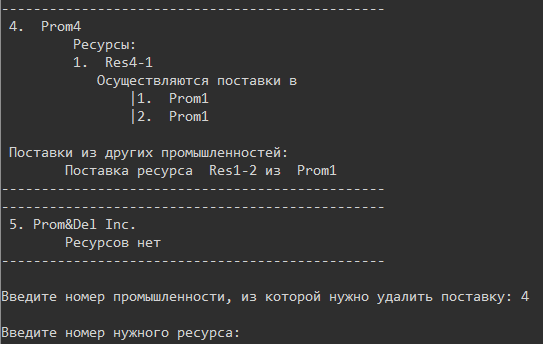
Пользователю выведется список существующей базы и представлена возможность выбора, какая промышленность должна поставлять ресурс, после чего программа запросит номер поставляемого ресурса:

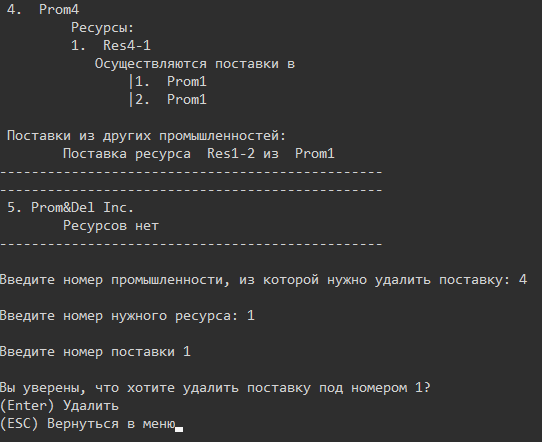
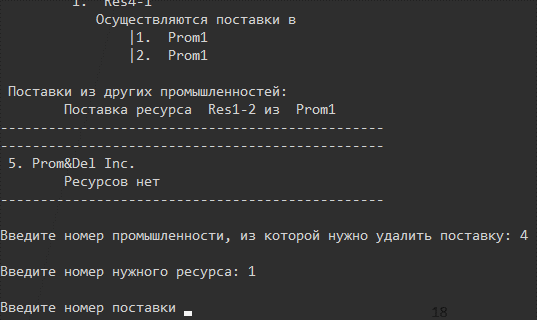
И номер промышленности, куда этот ресурс поставлять:

После чего выведется полученный результат (рис. №1)

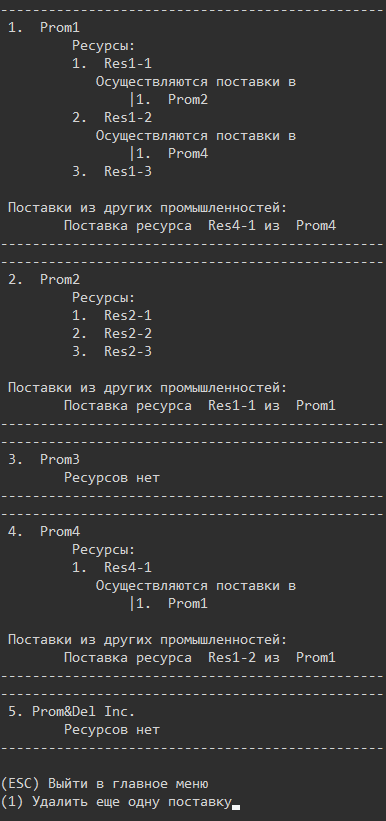
В свою очередь, для удаления поставки, выберем 6 пункт в главном меню и вновь увидим весь список , с возможностью ввода номера объекта, откуда собираемся удалить поставку (рис. №2)

 Рис. 1 Рис. 2

Далее введём номер промышленности:

Номер ресурса:

Номер поставки (рис. 3):

Подтвердим удаление (enter): Рис. 3

Заметим, что поставка №1 ресурса Res4-1 из промышленности Prom4 была успешно удалена!

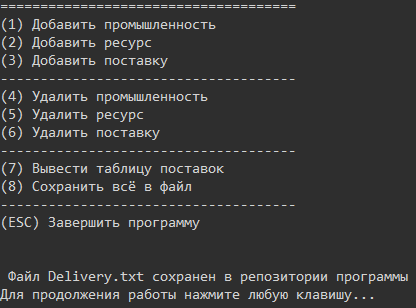
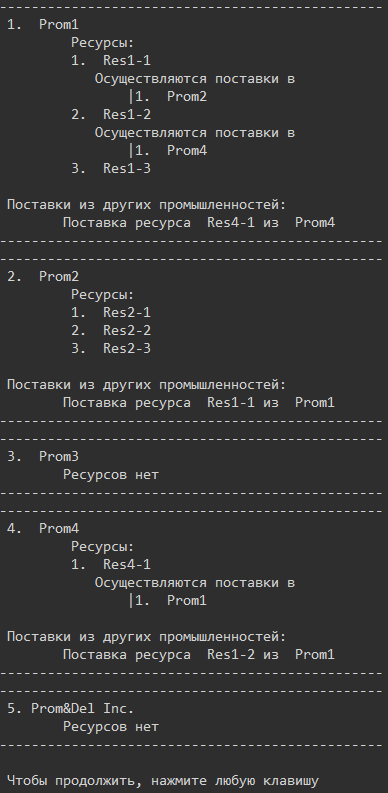
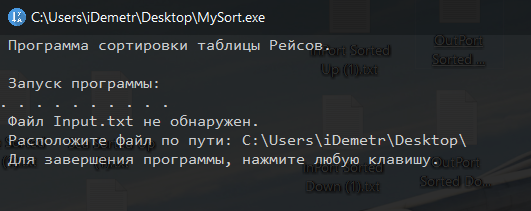
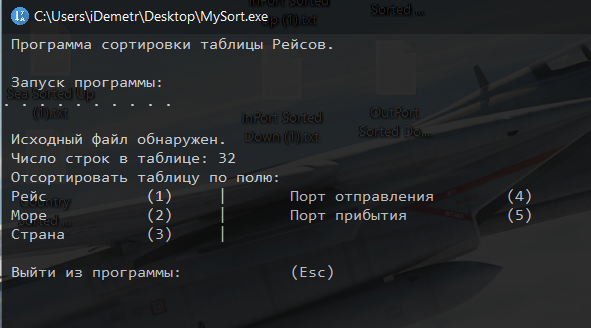
Чтобы посмотреть существующий список поставок можно выбрать 7 пункт главного меню, получим: (рис 4.)

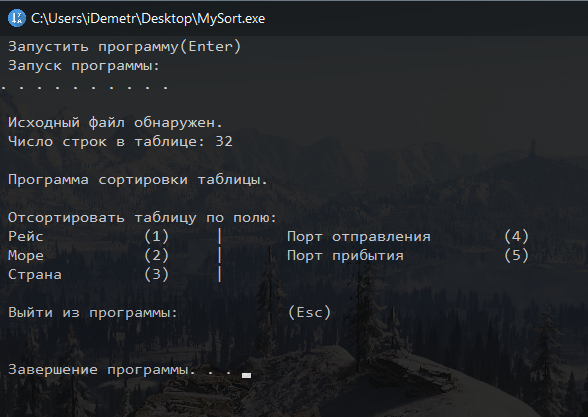
Рис. 5

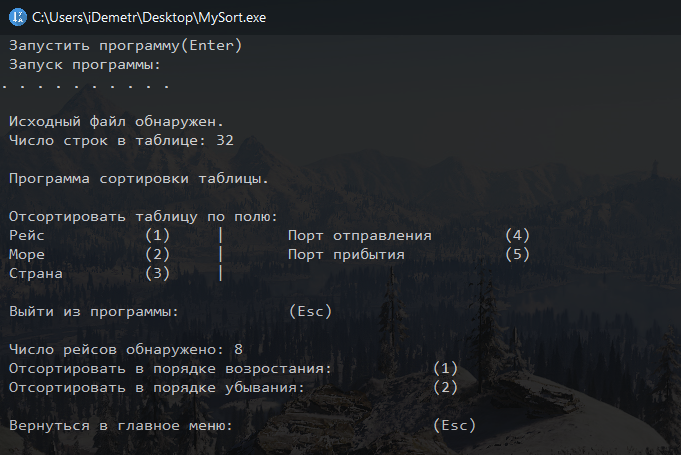
Рис. 4

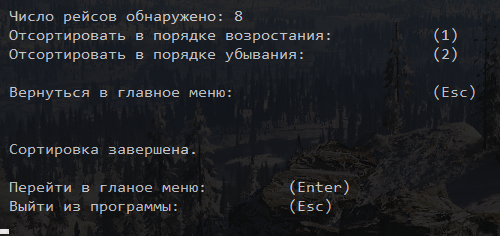
Или выберем 8 пункт главного меню, чтобы записать все записи в файл! (рис. 5)

Чтобы выйти из программы – жмём клавишу esc в главном меню.

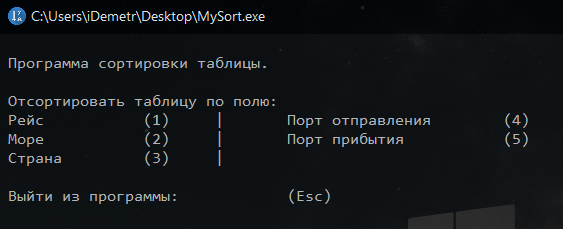
1. **Тестирование**
2. После запуска программы, если файл не был обнаружен, программа выводит соответствующее сообщение и завершает работу: 
3. Если же файл был обнаружен, программа выводит главное меню: 
4. При выборе варианта «Выйти из программы» (Esc) программа завершает работу:



1. Если выбрать вариант сортировки по рейсу (1), программа выводит число обнаруженных рейсов и предлагает выбрать направление сортировки:
2. При выборе варианта (1) или (2) программа создает файл с именем, соответствующее Параметру и направлению сортировки и выводит сообщение об успешной сортировке, и предлагает выйти или в меню или из программы:



1. Если выбрать переход в меню (Enter):



Тесты пунктов меню 2-5 соответствуют тестам IV - VI.

1. **Листинг  
   \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**