«Московский авиационный институт»   
(Национальный исследовательский университет)

Лабораторная работа по курсу

«Технологии программирования»

Сортировка данных методом сортировки выбором

Задание 2

Вариант 24

Выполнил:   
студент группы

 М7О-207С-19

Понкращенков Дмитрий

Проверил:

Барчев Николай Борисович

Москва

2021

**Содержание**

1. Задание. ………………………..………………………………………………………………………………….……...3
2. Псевдокод. ………………………….…….…………………………………………………………………….….…...4
3. Сведения о программной реализации.  …………………….………………………………………...12
4. Инструкция пользователя. …………………………….……………………………………………………...16
5. Тестирование. .………………………………………………………………….…………………………………...18
6. Листинг. ………………………………………………….……………………………………………………………...25
7. **Задание**
   1. Обеспечить адекватное задаче взаимодействие с пользователем и необходимые по смыслу задания проверки корректности данных;
   2. Обеспечить возможность сортировки по любому информационному элементу текстовой информации (полю), находящемуся в строках сортируемого файла **(по выбору пользователя);**
   3. Упорядочение при сортировке должно выполняться лексикографически с возможностью изменения его направления **(по выбору пользователя);**
   4. Результат **сортировки** каждый раз должен выводиться **в** **новый** **текстовый файл;** в файле должна содержаться информация о поле, по которому выполнялась сортировка, и о направлении сортировки;
   5. Информационные элементы (поля), по которым выполняется сортировка, в отсортированном файле должны выводиться в начале соответствующих строк;
   6. Учесть, что ограничение на размер входного файла отсутствует;
   7. **В обязательном порядке использовать языковые средства организации программных единиц;**
   8. При разработке использовать средства одной из реализаций языка программирования **C++;**
   9. **Сортировку выполнять методом выбора.**
8. **Псевдокод**

**Функция main:**

* Открыть входной файл Input.txt.
* Если файл открыт:
* Подсчитать количество строк в файле, для выделения динамической памяти.
* Пока не достигнут конец входного файла
* Считать строку.
* Вывести количество строк в файле.
* Объявить динамический двумерный массив для записи данных из файла.

//Считывание данных из файла и заполнение массива.

* Пока не достигнут конец входного файла.
* Пропустить первую строку – наименования столбцов.
* Считать название рейса.
* Если название рейса не прочерки, то у рейса есть Порт Отправления и Прибытия, и соответствующие им Моря и Страны.
* Считать:
  + - * Порт Отправления;
      * Страну Отправления;
      * Море Отправления;
      * Порт Прибытия;
      * Страну Прибытия;
      * Море Прибытия;
* Иначе: рейса нет, есть только порт и соответствующее ему море и страна.
* Считать:
  + - * Порт Отправления;
      * Страну Отправления;
      * Море Отправления;
* Цикл главного меню.
* Вывести главное меню программы.
* Считать символ.
* Если выбрана сортировка по рейсам (1):
* Спросить направление сортировки.
* Задать соответствующее название файла.
* Вызвать соответствующую функцию сортировки.
* Повторить для остальных пунктов меню (2-5).
* Если выбран пункт выхода из программы:
* Завершить программу.
* Если сортировка была выполнена.
* Предложить пользователю выйти в меню или выйти из программы.
* Освободить память от динамического массива
* Завершить программу.
* Иначе, если файл открыть не удалось:
* Предупредить пользователя.

**Функция sortUp** //Функция сортировки столбца по возрастанию.

* Записать в локальный массив параметров сортируемый столбец.
* Записать локальный массив индексов.

//Сортировка Локального Массива.

* Найти минимальный элемент в массиве.
* Поменять местами с первым элементом в массиве элементов.
* Поменять местами с первым индексом элемента в массиве индексов.
* Найти следующий максимальный элемент в неотсортированной части массива.
* Вызвать функцию вывода, передать отсортированный массив индексов.
* Очистить память от локальных массивов.

**Функция sortDown** //Функция сортировки столбца по убыванию.

* Записать в локальный массив параметров сортируемый столбец.
* Записать локальный массив индексов.

//Сортировка Локального Массива.

* Найти максимальный элемент в массиве.
* Поменять местами с первым элементом в массиве элементов.
* Поменять местами с первым индексом элемента в массиве индексов.
* Найти следующий максимальный элемент в неотсортированной части массива.
* Вызвать функцию вывода, передать отсортированный массив индексов.
* Очистить память от локальных массивов.

**Функция vivod** //Функция вывода отсортированного массива.

* Открыть файл с именем, соответствующий параметру и направлению сортировки.
* Если имя файла соответствует Рейсам:
* Вызвать функцию проверки существующих имен файлов.
* Вывести заголовок таблицы.
* Считать в локальный счетчик индекс, полученный из массива индексов.
* Если имя рейса не прочерки:
* Вывести в файл:
  + - * Массив Рейсов;
      * Массив Портов Отправления;
      * Массив Стран Отправления;
      * Массив Морей Отправления;
      * Массив Портов Прибытия;
      * Массив Стран Прибытия;
      * Массив Морей Прибытия;
* Если имя файла соответствует Портам Отправления:
* Вывести заголовок таблицы.
* Считать в локальный счетчик индекс, полученный из массива индексов.
* Если имя Порта Отправления не прочерки:
* Вывести в файл:
  + - * Массив Портов Отправления;
      * Массив Рейсов;
      * Массив Стран Отправления;
      * Массив Морей Отправления;
      * Массив Портов Прибытия;
      * Массив Стран Прибытия;
      * Массив Морей Прибытия;
* Если имя файла соответствует Портам Прибытия:
* Вывести заголовок таблицы.
* Считать в локальный счетчик индекс, полученный из массива индексов.
* Если имя Порта Прибытия не прочерки:
* Вывести в файл:
  + - * Массив Портов Прибытия;
      * Массив Рейсов;
      * Массив Стран Прибытия;
      * Массив Морей Прибытия;
      * Массив Портов Отправления;
      * Массив Стран Отправления;
      * Массив Морей Отправления;
* Закрыть файл.

**Функция sortUp2** //Функция сортировки столбца Морей и Стран по возрастанию.

//Запись в Локальный Массив.

* Считать в локальный массив параметров столбец Отправления Моря/Страны.
* Записать локальный массив индексов.
* Если столбец прибытия Моря/Страны не пуст:
* Следующим элементом в массив параметров записать столбец Прибытия Моря/Страны.
* Следующим элементом в массив индексов записать следующий индекс.

//Сортировка Локального Массива.

* Найти минимальный элемент в массиве.
* Поменять местами с первым элементом в массиве элементов.
* Поменять местами с первым индексом элемента в массиве индексов.
* Найти следующий максимальный элемент в неотсортированной части массива.
* Вызвать функцию вывода, передать отсортированный массив индексов.
* Очистить память от локальных массивов.

**Функция sortDown2** //Функция сортировки столбца Морей и Стран по убыванию.

//Запись в Локальный Массив.

* Считать в локальный массив параметров столбец Отправления Моря/Страны.
* Записать локальный массив индексов.
* Если столбец прибытия Моря/Страны не пуст:
* Следующим элементом в массив параметров записать столбец Прибытия Моря/Страны.
* Следующим элементом в массив индексов записать следующий индекс.

//Сортировка Локального Массива.

* Найти максимальный элемент в массиве.
* Поменять местами с первым элементом в массиве элементов.
* Поменять местами с первым индексом элемента в массиве индексов.
* Найти следующий максимальный элемент в неотсортированной части массива.
  + Вызвать функцию вывода, передать отсортированный массив индексов.
  + Очистить память от локальных массивов.

**Функция vivod2** //Функция вывода отсортированного массива.

* Открыть файл с именем, соответствующий параметру и направлению сортировки.
* Вызвать функцию проверки существующих имен файлов.
* Если имя файла соответствует Морям:
* Считать в локальный массив1 – массив Морей Отправления.
* Считать в локальный массив2 – массив Морей Прибытия.
* Вывести заголовок таблицы.
* Считать в локальный счетчик индекс, полученный из массива индексов.
* Если параметр из отсортированного массива совпадает с параметром локального массива1 и он не равен прочеркам:
  + Вывести Море отправления Рейса.
  + Вывести Рейс.
  + Вывести Порт Отправления Рейса.
  + Вывести Страну Отправления Рейса.
  + Обнулить в локальном массиве1 данный параметр.
* Если параметр из отсортированного массива совпадает с параметром локального массива2 и он не равен прочеркам:
  + Вывести Море прибытия Рейса.
  + Вывести Рейс.
  + Вывести Порт Прибытия Рейса.
  + Вывести Страну Прибытия Рейса.
  + Обнулить в локальном массиве2 данный параметр.
* Если имя файла соответствует Странам:
* Считать в локальный массив1 – массив Стран Отправления.
* Считать в локальный массив2 – массив Стран Прибытия.
* Вывести заголовок таблицы.
* Считать в локальный счетчик индекс, полученный из массива индексов.
* Если параметр из отсортированного массива совпадает с параметром локального массива1 и он не равен прочеркам:
  + Вывести Страну Отправления Рейса.
  + Вывести Рейс.
  + Вывести Порт Отправления Рейса.
  + Вывести Море Отправления Рейса.
* Если параметр из отсортированного массива совпадает с параметром локального массива2 и он не равен прочеркам:
  + Вывести Страну Прибытия Рейса.
  + Вывести Рейс.
  + Вывести Порт Прибытия Рейса.
  + Вывести Море Прибытия Рейса.
* Закрыть файл.

**Функция ifFileDetected** //Функция увеличения номера файла в его названии.

* Если файл с текущим названием уже существует:
* Найти позицию в строке номера файла.
* Считать номер файла.
* Увеличить номер файла на 1.
* Заменить старый номер файла на новый.
* Проверить имя файла еще раз.
* Если файл с увеличенным номером файла не обнаружен:
* Вернуть это имя файла в программу.

1. **Сведения о программной реализации:**

Версия системы программирования:   
Microsoft Visual Studio Community 2019 Version 16.9.5  
  
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 | * 82 кб ОЗУ, рекомендуется 164 кб ОЗУ |

**Описание входных и выходных данных:**

Входными данными будет файл, полученный в результате работы второй программы задания 1, расположенный в одной папке с программой:

Input.txt

Выходными данными будут текстовые файлы: *имена файлов задаются автоматически в соответствии с выбранным параметром сортировки и его направлением, содержащие данные, полученные во входном файле, но отсортированные по выбранному параметру и направлении.*

**Особые указания:**

|  |  |
| --- | --- |
| Диагностические сообщения | Причина возникновения |
| «Файл Input.txt не обнаружен. Расположите файл по пути: C:Users/User/Desktop/» | Входной файл не лежит в одной папке с программой. |

* Выходные файлы будут автоматически созданы в одной папке с программой.
* Для корректного просмотра выходных файлов рекомендуется открывать текстовые файлы при помощи NotePad++, c установленным размером табуляционных отступов в 8 единиц.

**Программные единицы:**

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

Функция *SortUp (int countStlb, int nomberStlb, string\*\* array, string fileName).*

Назначение: Сортирует по столбцу 1 параметр (Рейс, Порт Отправления, Порт Прибытия) по возростанию.

Тип: *void*;

Входные параметры: «*countStlb*» - Кол-во эл-тов в сортируемом столбце. Тип:*int*;

«*nomberStlb*» - Номер сортируемого столбца. Тип: *int*;

«*array*» - Сортируемый массив. Тип: *string\*\**;

«*fileName*» - Имя файла. Тип: *string*;

Возвращаемое значение отсутствует.

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

Функция *sortDown (int countStlb, int nomberStlb, string\*\* array, string fileName).*

Назначение: Сортирует по столбцу 1 параметр (Рейс, Порт Отправления, Порт Прибытия) по убыванию.

Тип: *void*;

Входные параметры: «*countStlb*» - Кол-во эл-тов в сортируемом столбце. Тип:*int*;

«*nomberStlb*» - Номер сортируемого столбца. Тип: *int*;

«*array*» - Сортируемый массив. Тип: *string\*\**;

«*fileName*» - Имя файла. Тип: *string*;

Возвращаемое значение отсутствует.

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

Функция *vivod (int countStlb, string\*\* array, int\* arrayIndex, string fileName).*

Назначение: Выводит массив по правилам сортировки.

Тип: *void*;

Входные параметры: «*countStlb*» - Кол-во эл-тов в сортируемом столбце. Тип:*int*;

«*array*» - Сортируемый массив. Тип: *string\*\**;

«*arrayIndex*» - массив отсортированных индексов. Тип: *int\**;

«*fileName*» - Имя файла. Тип: *string*;

Возвращаемое значение отсутствует.

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

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

Функция *sortUp2 (int countStlb, int nomberStlb, string\*\* array, string fileName)*

Назначение: Сортирует по столбцу 2 параметра(Страну Отправления и Прибытия / Море Отправления / Прибытия) по возростанию.

Тип: *void*;

Входные параметры: «*countStlb*» - Кол-во эл-тов в сортируемом столбце. Тип:*int*;

«*nomberStlb*» - Номер сортируемого столбца. Тип: *int*;

«*array*» - Сортируемый массив. Тип: *string\*\**;

«*fileName*» - Имя файла. Тип: *string*;

Возвращаемое значение отсутствует.

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

Функция *sortDown2 (int countStlb, int nomberStlb, string\*\* array, string fileName)*

Назначение: Сортирует по столбцу 2 параметра(Страну Отправления и Прибытия / Море Отправления / Прибытия) по убыванию.

Тип: *void*;

Входные параметры: «*countStlb*» - Кол-во эл-тов в сортируемом столбце. Тип:*int*;

«*nomberStlb*» - Номер сортируемого столбца. Тип: *int*;

«*array*» - Сортируемый массив. Тип: *string\*\**;

«*fileName*» - Имя файла. Тип: *string*;

Возвращаемое значение отсутствует.

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

Функция *vivod2* (*int countStlb, string\*\* array, int\* arrayIndex, string\* arrayStr, string fileName).*

Назначение: Выводит массив в файл по правилам сортировки.

Тип: *void*;

Входные параметры: «*countStlb*» - Кол-во эл-тов в сортируемом столбце. Тип:*int*;

«*array*» - Сортируемый массив. Тип: *string\*\**;

«*arrayIndex*» - массив отсортированных индексов. Тип: *int\**;

«*arrayStr*» -массив отсортированных параметров. Тип: *int\*;*

«*fileName*» - Имя файла. Тип: *string*;

Возвращаемое значение отсутствует.

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

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

Функция ***Loading*** (*int MiliSek,int Kolich\_Tochek*).

Назначение: Анимация загрузки меню.

Тип: *void*;

Входные параметры: «*MiliSek*» - кол-во миллисекунд задержки, Тип: *int;*

«*Kolich\_Tochek*» - кол-во печатаемых точек, Тип: *int*;

Возвращаемое значение отсутствует.

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

Функция ***Start*** *().*

Назначение: Анимация инициализации программы.

Тип: *void;*

Входные параметры: отсутствуют;

Возвращаемое значение отсутствует.

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

Функция ***Tab*** (*string word*)*.*

Назначение: корректировка вывода в файл элементов таблицы.

Тип: *string*;

Входные параметры: «*word*»– слово, относительно которого происходит последующая табуляция;

Возвращаемое значение: “\t\t|\t” или “\t|\t” в зависимости какой длины слово.

Тип: *string*;

Функция ***ifFileDetected*** (*string* *oldFileName*).

Назначение: Создание файла с уникальным названием.

Тип: string;

Входные параметры: «*oldFileName*» - Имя файла до корректирования. Тип: *string;*

Возвращаемое значение: «*oldFileName*» - Корректное имя файла. Тип: *string;*

Функция ***message ().***

Назначение: Выводит повторяющийся текст.

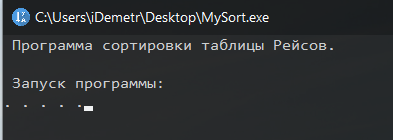
Тип: void;

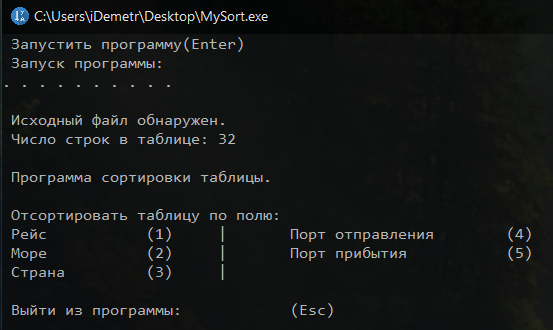
Входные параметры: отсутствуют;

Возвращаемое значение отсутствует;

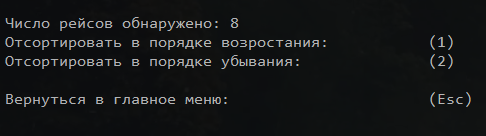
1. **Инструкция пользователя:**

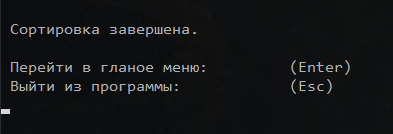
После запуска программы на экране монитора появляется анимация инициализации программы:

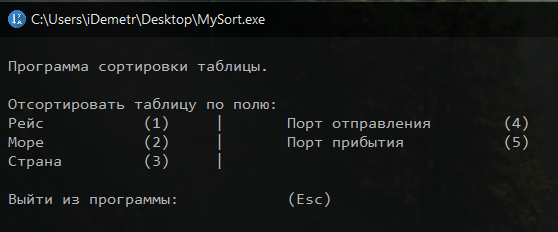


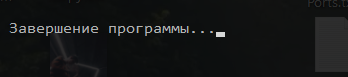
После проигрывания анимации на экране монитора появляются сообщения, информирующие, что исходный файл был обнаружен, найденное число строк во входном файле и меню с выбором действий:

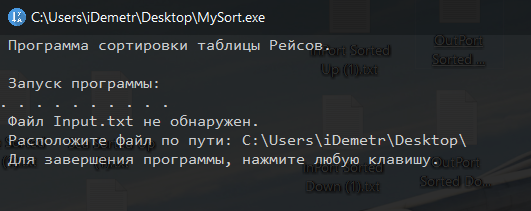
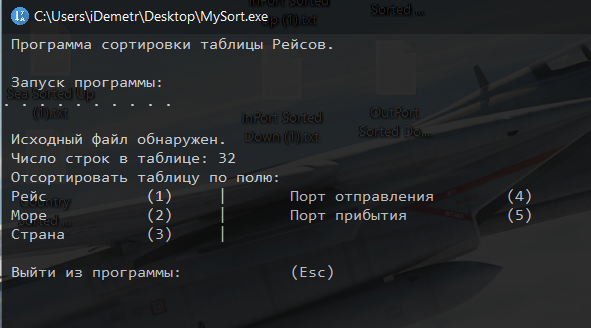
Выбор пунктов осуществляется с помощью ввода нужного номера пункта с клавиатуры.

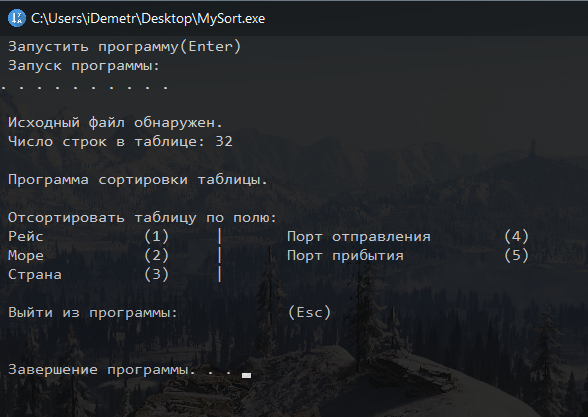
После выбора пользователем, например, первого пункта меню на экране появится сообщение о количестве найденных рейсов, а также меню с выбором направления сортировки:

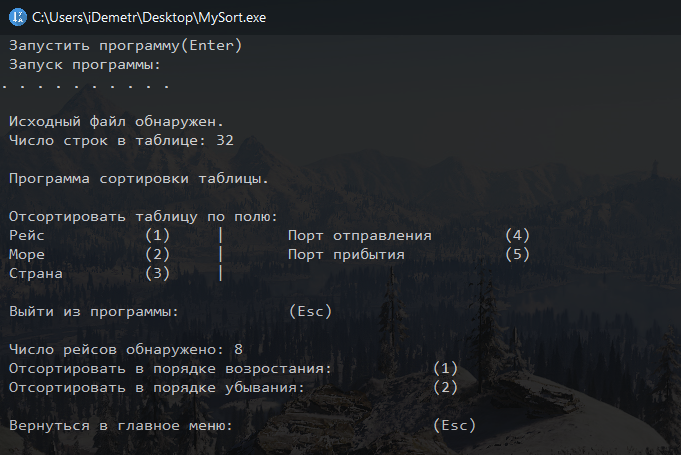
После выбора пользователем, например, первого пункта меню на экране появится сообщение о успешной сортировке и меню с выбором завершением программы или переходом в главное меню для продолжения сортировки:

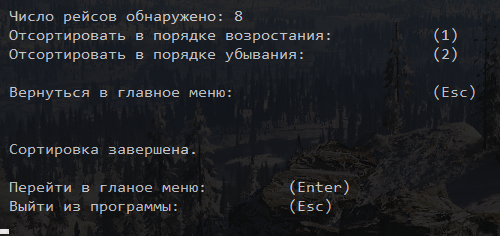
При выборе пункта «Перейти в главное меню: (Enter)» будет произведена очистка экрана и переход в главное меню:

Все последующие пункты (2-5) меню будут иметь такую же структуру, кроме выбора пункта «Выйти из программы (Esc)».   
Будет проигранна анимация завершения программы и последующий выход из нее:

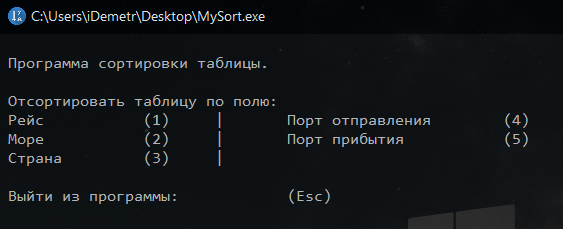
1. **Тестирование**
2. После запуска программы, если файл не был обнаружен, программа выводит соответствующее сообщение и завершает работу: 
3. Если же файл был обнаружен, программа выводит главное меню: 
4. При выборе варианта «Выйти из программы» (Esc) программа завершает работу:



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:



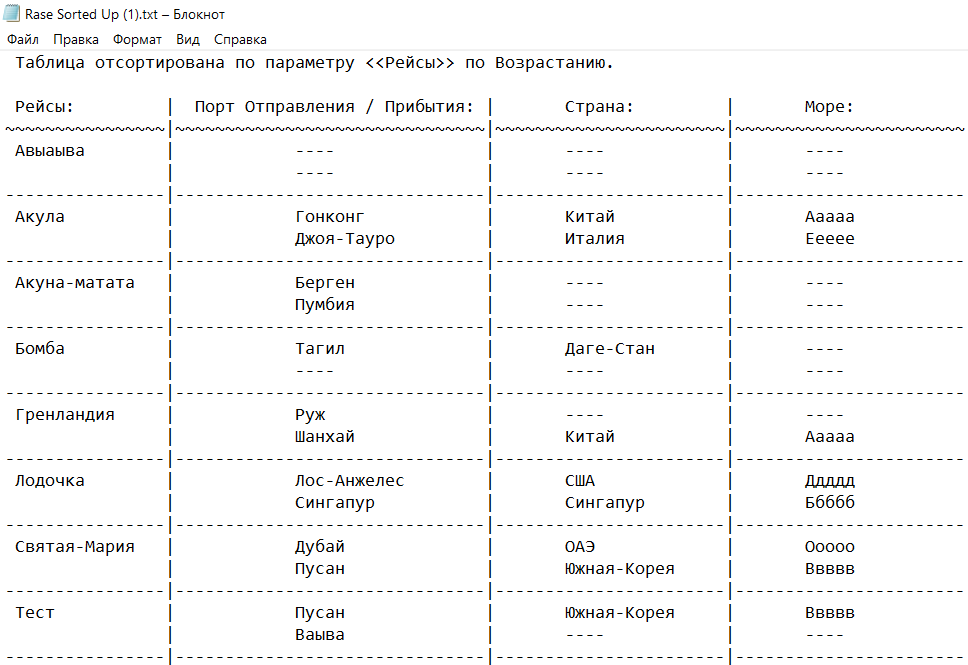
1. Если выбрать переход в меню (Enter):

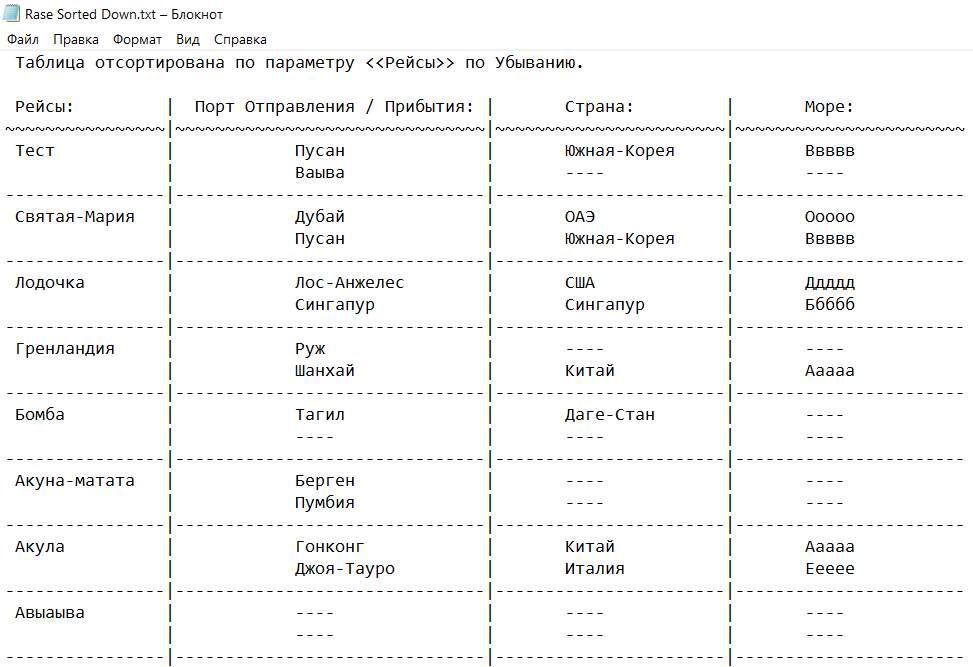


Тесты пунктов меню 2-5 соответствуют тестам IV - VI.

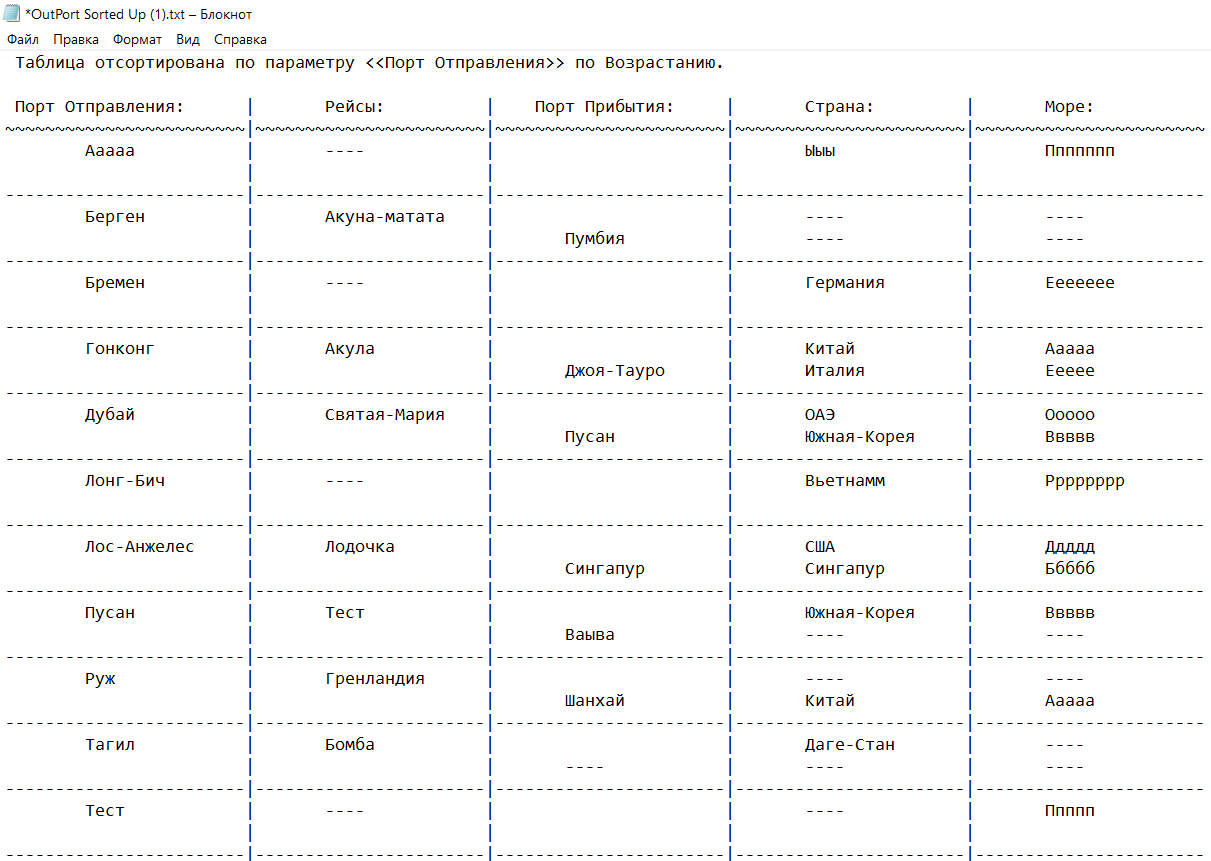
В результате работы программы, могут быть сформированы следующие файлы:

Файл сортировки Рейсов по возрастанию:

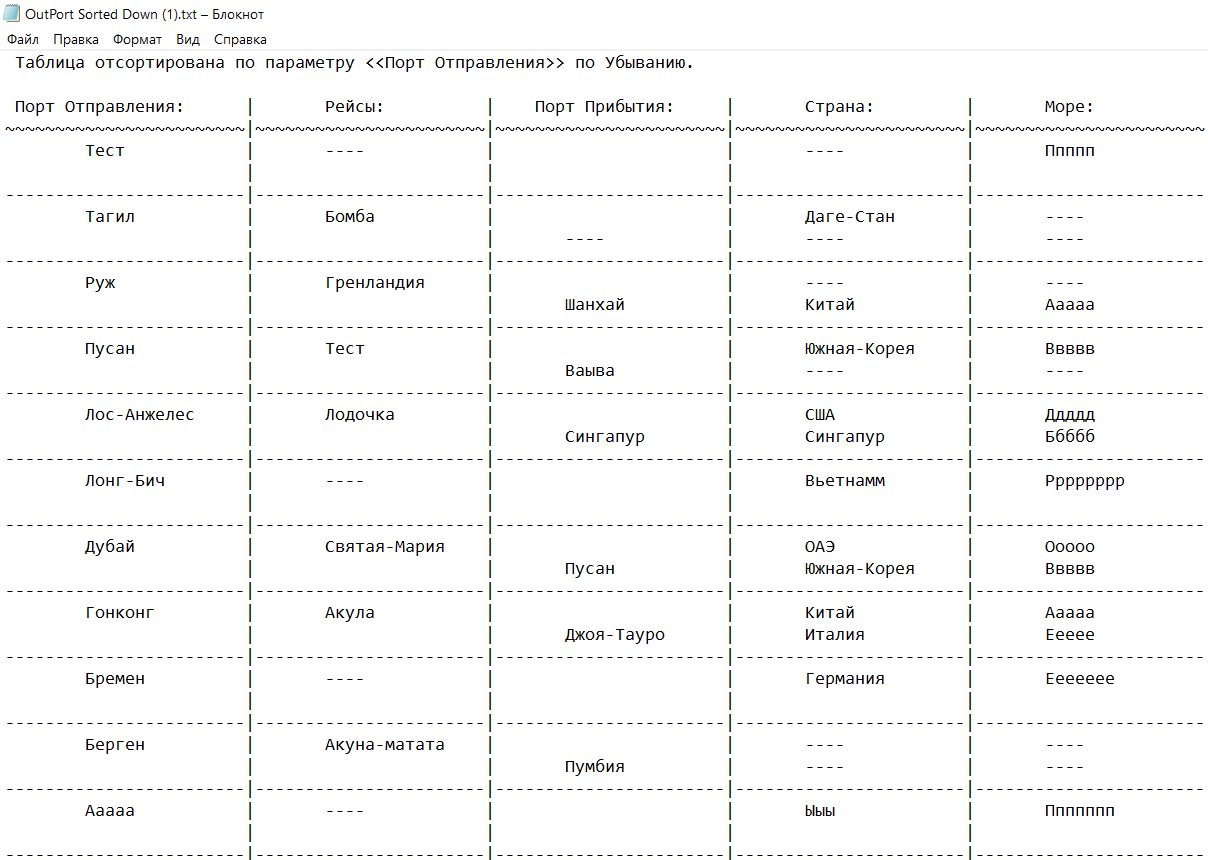


Файл сортировки Рейсов по убыванию: 

Файл сортировки Портов Отправления по возрастанию:



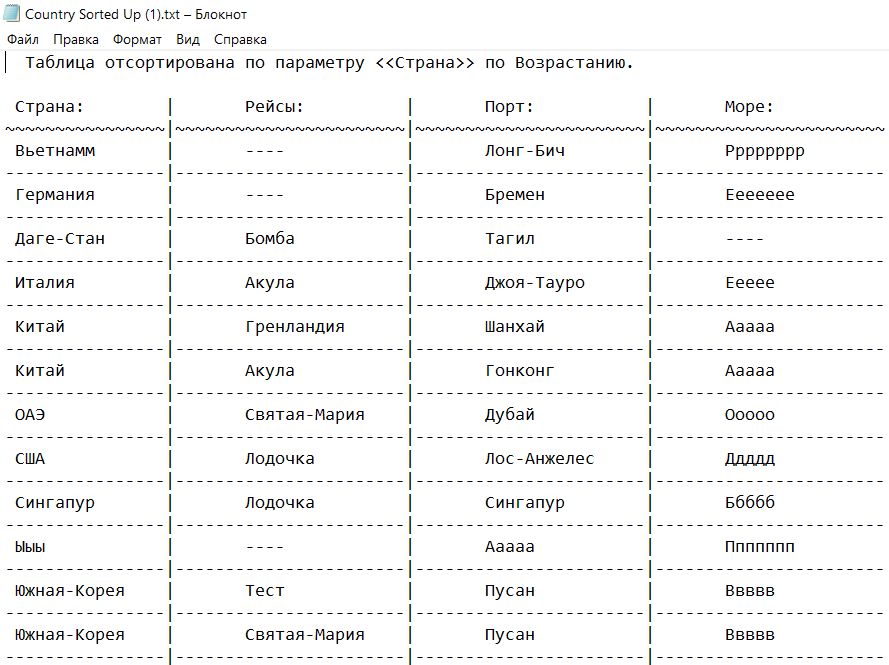
Файл сортировки Портов Отправления по убыванию:



Файл сортировки Портов Прибытия по возрастанию: 

Файл сортировки Портов Прибытия по убыванию:



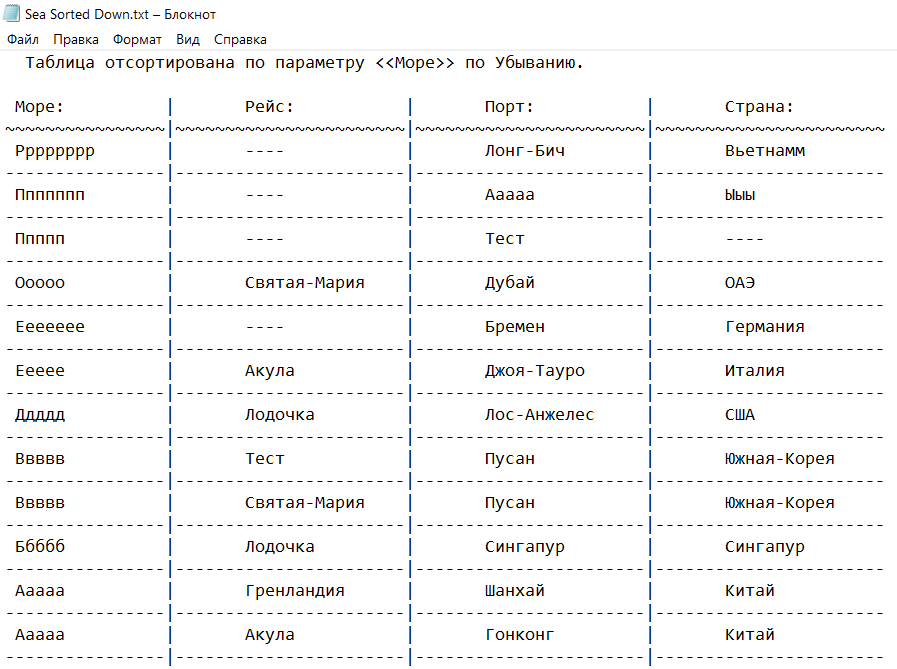
Файл сортировки Стран по возрастанию: 

Файл сортировки Стран по убыванию:



Файл сортировки Морей по возрастанию:



Файл сортировки Морей по убыванию:

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

**Menu.Cpp** //Главное меню и ветвление сортировки.

1. #include "MySort.h" //Заголовочный файл функций вывода и сортировки по одному аргументу.
2. #include "MySort2xStr.h" //Заголовочный файл функций вывода и сортировки по двум аргументам.
3. using namespace std;
4. void massage(); //Прототип функции.
5. int main()
6. {
7. setlocale(LC\_ALL, "Russian");
8. SetConsoleCP(1251); //Русский язык в консоли.
9. SetConsoleOutputCP(1251); //Русский язык в файле.
10. int i(0); //Счетчик.
11. int j(0); //Счетчик по строкам таблицы.
12. int ch1(0); //Буфер ввода.
13. int ch(0); //Буфер ввода.
14. int countStr(0); //Число строк в таблице.
15. int countRase(0); //Число рейсов.
16. int countPort(0); //Число портов.
17. int nomberStr(0); //Номер строки в настоящий момент.
18. bool Flag1 = true; //Логическая переменная для перехода в главное меню программы.
19. string str;
20. string bufer;
21. string InputFile("Input.txt"); //Входной файл Рейсов до Сортировки.
22. string FileName; //Имя выходного файла.
23. WCHAR PathName[MAX\_PATH]; //Путь к расположению программы.
24. GetModuleFileName(NULL, PathName, sizeof(PathName));
25. PathName[wcslen(PathName) - 10] = 0; //Игнорировать вывод имени программы в пути.
26. cout << " Программа сортировки таблицы Рейсов.\n\n";
27. Start(); //Анимация запуска программы.
28. ifstream f1(InputFile); //Открываем исходный файл.
29. if (f1.is\_open()) //Проверка наличия исходного файла.
30. {
31. cout << "\n\n Исходный файл обнаружен.\n";
32. //Подсчет колличества строк в таблице.
33. while (!f1.eof()) //Подсчет количества строк в таблице.
34. {
35. getline(f1, str);
36. countStr++;
37. }
38. cout << " Число строк в таблице: " << countStr << endl;
39. //Объявления динамического массива для заполнения данными из входного файла.
40. string\*\* array = new string \* [7]; //Где 7 – кол-во столбцов в таблице.
41. for (i = 0; i < 7; i++)
42. array[i] = new string[countStr];
43. f1.seekg(0); //Перевод маркера текущей позиции в начало файла.
44. //Главное меню.
45. while (Flag1 == true) //Цикл главного меню.
46. {
47. cout << " Отсортировать таблицу по полю:\n";
48. cout << " Рейс\t\t(1)\t|\tПорт отправления\t(4)\n Море\t\t(2)\t|\tПорт прибытия\t\t(5)\n Страна\t\t(3)\t|\n";
49. cout << "\n Выйти из программы:\t\t(Esc)\n\n";
50. //Диалог до сортировки.
51. do
52. {
53. ch = \_getch();
54. } while (ch != '1' && ch != '2' && ch != '3' && ch != '4' && ch != '5' && ch != 27);
55. i = 0;
56. //Считывание из входного файла в массив.
57. while (!f1.eof()) //Пока не достигнут конец файла.
58. { //Сканировать каждую строку в таблице.
59. i++; //Какая строка сейчас активна.
60. if (i > 5) //Пропуск первых Шести слов в таблице.
61. {
62. f1 >> bufer; //Считать имя рейса из файла. //Наименование рейса. [0]
63. array[0][nomberStr] = bufer;
64. if (bufer == "----") //Если Рейса нет, а есть только порты.
65. {
66. f1 >> array[1][nomberStr]; //Порт отправления.
67. f1 >> array[2][nomberStr]; //Страна отправления.
68. f1 >> array[3][nomberStr]; //Море отправления.
69. countPort++;
70. }
71. else //Если имя рейса существует.
72. {
73. f1 >> array[1][nomberStr]; //Порт отправления. [1]
74. f1 >> array[2][nomberStr]; //Страна отправления. [2]
75. f1 >> array[3][nomberStr]; //Море отправления. [3]
76. f1 >> array[4][nomberStr]; //Порт прибытия. [4]
77. f1 >> array[5][nomberStr]; //Страна прибытия. [5]
78. f1 >> array[6][nomberStr]; //Море порта. [6]
79. countRase++;
80. }
81. nomberStr++;
82. }
83. else
84. {
85. f1 >> bufer;
86. }
87. }
88. if (ch == '1') //Сортировка по имени рейса.
89. {
90. cout << " Число рейсов обнаружено: " << countRase << endl;
91. massage();
92. do
93. {
94. ch1 = \_getch();
95. } while (ch1 != 49 && ch1 != 50 && ch1 != 27);
96. if (ch1 == '1') //Сортировка по возрастанию.
97. {
98. FileName = "Rase Sorted Up (1).txt";
99. sortUp(nomberStr, 0, array, FileName);
100. Flag1 = false;
101. }
102. if (ch1 == '2') //Сортировка по убыванию.
103. {
104. FileName = "Rase Sorted Down (1).txt";
105. sortDown(nomberStr, 0, array, FileName);
106. Flag1 = false;
107. }
108. if (ch1 == 27)
109. {
110. system("cls");
111. }
112. }
113. if (ch == '4') //Сортировка по порту отправления.
114. {
115. cout << " Число Портов Отправления обнаружено: " << nomberStr << endl;
116. massage();
117. do
118. {
119. ch1 = \_getch();
120. } while (ch1 != 49 && ch1 != 50 && ch1 != 27);
121. if (ch1 == '1') //Сортировка по возрастанию.
122. {
123. FileName = "OutPort Sorted Up (1).txt";
124. sortUp(nomberStr, 1, array, FileName);
125. Flag1 = false;
126. }
127. if (ch1 == '2') //Сортировка по убыванию.
128. {
129. FileName = "OutPort Sorted Down (1).txt";
130. sortDown(nomberStr, 1, array, FileName);
131. Flag1 = false;
132. }
133. if (ch1 == 27)
134. {
135. system("cls");
136. }
137. }
138. if (ch == '2') //Сортировка по морям.
139. {
140. cout << " Число Морей обнаружено: " << nomberStr + countRase << endl;
141. massage();
142. int countSea(0);
143. do
144. {
145. ch1 = \_getch();
146. } while (ch1 != 49 && ch1 != 50 && ch1 != 27);
147. countSea = countRase + nomberStr;
148. if (ch1 == '1') //Сортировка по возрастанию.
149. {
150. FileName = "Sea Sorted Up (1).txt";
151. sortUp2(countSea, 3, array, FileName);
152. Flag1 = false;
153. }
154. if (ch1 == '2') //Сортировка по убыванию.
155. {
156. FileName = "Sea Sorted Down (1).txt";
157. sortDown2(countSea, 3, array, FileName);
158. Flag1 = false;
159. }
160. if (ch1 == 27)
161. {
162. system("cls");
163. }
164. }
165. if (ch == '5') //Сортировка по порту прибытиям.
166. {
167. cout << " Число Портов Прибытия обнаружено: " << countRase << endl;
168. massage();
169. do
170. {
171. ch1 = \_getch();
172. } while (ch1 != 49 && ch1 != 50 && ch1 != 27);
173. if (ch1 == '1') //Сортировка по возрастанию.
174. {
175. FileName = "InPort Sorted Up (1).txt";
176. sortUp(nomberStr, 4, array, FileName);
177. Flag1 = false;
178. }
179. if (ch1 == '2') //Сортировка по убыванию.
180. {
181. FileName = "InPort Sorted Down (1).txt";
182. sortDown(countRase, 4, array, FileName);
183. Flag1 = false;
184. }
185. if (ch1 == 27)
186. {
187. system("cls");
188. }
189. }
190. if (ch == '3') //Сортировка по странам.
191. {
192. cout << " Число Стран обнаружено: " << nomberStr + countRase << endl;
193. massage();
194. int countCountry(0);
195. do
196. {
197. ch1 = \_getch();
198. } while (ch1 != 49 && ch1 != 50 && ch1 != 27);
199. countCountry = countRase + nomberStr;
200. if (ch1 == '1') //Сортировка по возрастанию.
201. {
202. FileName = "Country Sorted Up (1).txt";
203. sortUp2(countCountry, 2, array, FileName);
204. Flag1 = false;
205. }
206. if (ch1 == '2') //Сортировка по убыванию.
207. {
208. FileName = "Country Sorted Down (1).txt";
209. sortDown2(countCountry, 2, array, FileName);
210. Flag1 = false;
211. }
212. if (ch1 == 27)
213. {
214. system("cls");
215. }
216. }
217. if (ch == 27)
218. {
219. cout << "\n Завершение программы";
220. Loading(500, 4); //Анимация загрузки.
221. return 1;
222. }
223. //Диалог После сортировки.
224. if (ch1 != 27)
225. {
226. printf("% ws\n", PathName);
227. cout << "\n\n Перейти в гланое меню:\t\t(Enter)\n Выйти из программы:\t\t(Esc)\n";
228. do
229. {
230. ch = \_getch();
231. } while (ch != 13 && ch != 27);
232. if (ch == 27) //Выход из программы.
233. {
234. cout << "\n Завершение программы";
235. Loading(500, 4); //Анимация завершения программы.
236. return 1;
237. }
238. else
239. if (ch == 13) //Переход в главное меню.
240. {
241. system("cls");
242. Flag1 = true;
243. }
244. }
245. }
246. //Освобождение памяти.
247. for (i = 0; i < 7; i++)
248. {
249. delete[] array[i];
250. }
251. delete[] array;
252. }
253. else
254. {
255. cout << "\n Файл Input.txt не обнаружен по пути: ";
256. printf("% ws\n", PathName);
257. cout << "Для завершения программы, нажмите любую клавишу.";
258. \_getch();
259. }
260. cout << "\n Завершение программы";
261. Loading(500, 4); //Анимация Завершения программы.
262. return 0;
263. }
264. void massage () //Функция вывода сообщений.
265. {
266. cout << " Отсортировать в порядке возрастания:\t\t(1)\n";
267. cout << " Отсортировать в порядке убывания: \t(2)\n\n";
268. cout << " Вернуться в главное меню:\t\t\t(Esc)\n\n";
269. }

**MySort.h** //Сортировка 1 аргумента и вывод таблицы в файл.

1. #include "Start.h" //Анимация инициализации программы.
2. #include <cstdio>
3. #include <string>
4. #include <iomanip>
5. #include <fstream>
6. #include "Tabulation on Tabl.h" //Функция табулирования в столбцах файла.
7. using namespace std;
8. //Функция вывода отсортированной таблицы.
9. void vivod(int countStlb, string\*\* array, int\* arrayIndex, string fileName)
10. //countStlb – номер сортируемого столбца.
11. //Array - сортируемый массив.
12. //arrayIndex – Массив индексов отсортированных элементов.
13. //arrayStr - Массив отсортированных строк.
14. //fileName - Имя выходного файла.
15. {
16. int str(0), stlb(0), i(0); //Счетчик.
17. fileName = ifFileDetected(fileName); //Замена на новое имя файла.
18. ofstream f2(fileName); //Открыть файл для заполнения.
19. //Вывод в файл Рейсов.
20. if (!fileName.find("Rase")) //Если встречен файл Рейсов.
21. {
22. //Вывод заголовка в файл.
23. {
24. if (fileName.find("Up") != string::npos) //Если сортировка по возрастанию.
25. {
26. f2 << " Таблица отсортирована по параметру <<Рейсы>> по Возрастанию.\n\n";
27. }
28. else //Если сортировка по убыванию.
29. {
30. f2 << " Таблица отсортирована по параметру <<Рейсы>> по Убыванию.\n\n";
31. }
32. f2 << " Рейсы:\t\t| Порт Отправления / Прибытия: |\tСтрана:\t\t|\tМоре:\n"; //Вывести 1-м столбец Рейсов.
33. f2 << "~~~~~~~~~~~~~~~~|~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~|~~~~~~~~~~~~~~~~~~~~~~~|~~~~~~~~~~~~~~~~~~~~~~~\n";
34. }
35. //Вывод в файл.
36. while (i < countStlb)
37. {
38. str = arrayIndex[i]; //Взятие индекса из массива индексов.
39. if (array[0][str] != "----")
40. {
41. f2 << " " << array[0][str] << tab(array[0][str]) << " "; //Вывод 1-м столбец Рейсов.
42. f2 << array[1][str] << tab3(array[1][str]);
43. f2 << array[2][str] << tab2(array[2][str]);
44. f2 << array[3][str] << "\n\t\t|\t ";
45. f2 << array[4][str] << tab3(array[4][str]);
46. f2 << array[5][str] << tab2(array[5][str]);
47. f2 << array[6][str];
48. f2 << "\n----------------|-------------------------------|-----------------------|-----------------------\n";
49. }
50. i++;
51. }
52. }
53. //Вывод в файл Портов Отправления.
54. if (!fileName.find("OutPort")) //Если встречен файл Портов Отправления.
55. {
56. //Вывод заголовка в файл.
57. {
58. if (fileName.find("Up") != string::npos) //Если сортировка по возростанию.
59. {
60. f2 << " Таблица отсортирована по параметру <<Порт Отправления>> по Возрастанию.\n\n";
61. }
62. else //Если сортировка по убыванию.
63. {
64. f2 << " Таблица отсортирована по параметру <<Порт Отправления>> по Убыванию.\n\n";
65. }
66. f2 << " Порт Отправления:\t|\tРейсы:\t\t| Порт Прибытия:\t|\tСтрана:\t\t|\tМоре:\n";
67. f2 << "~~~~~~~~~~~~~~~~~~~~~~~~|~~~~~~~~~~~~~~~~~~~~~~~|~~~~~~~~~~~~~~~~~~~~~~~|~~~~~~~~~~~~~~~~~~~~~~~|~~~~~~~~~~~~~~~~~~~~~~~\n";
68. }
69. //Вывод в файл.
70. while (i < countStlb)
71. {
72. str = arrayIndex[i]; //Взятие индекса из массива индексов.
73. if (array[1][str] != "----")
74. {
75. f2 << "\t" << array[1][str] << tab2(array[1][str]); //Вывод 1-м столбец Портов Отправления.
76. f2 << array[0][str] << tab2(array[0][str]) << "\t\t|\t";
77. f2 << array[2][str] << tab2(array[2][str]);
78. f2 << array[3][str] << "\n\t\t\t|\t\t\t|\t";
79. f2 << array[4][str] << tab2(array[4][str]);
80. f2 << array[5][str] << tab2(array[5][str]);
81. f2 << array[6][str] << "\n";
82. f2 << "------------------------|-----------------------|-----------------------|-----------------------|-----------------------\n";
83. }
84. i++;
85. }
86. }
87. //Вывод в файл Портов Прибытия.
88. if (!fileName.find("InPort")) //Если встречен файл Портов Прибытия.
89. {
90. //Вывод заголовка в файл.
91. {
92. if (fileName.find("Up") != string::npos) //Если сортировка по возрастанию.
93. {
94. f2 << " Таблица отсортирована по параметру <<Порт Прибытия>> по Возрастанию.\n\n";
95. }
96. else //Если сортировка по убыванию.
97. {
98. f2 << " Таблица отсортирована по параметру <<Порт Прибытия>> по Убыванию.\n\n";
99. }
100. f2 << " Порт Прибытия:\t|\tРейсы:\t\t| Порт Отправления:\t|\tСтрана:\t\t|\tМоре:\n";
101. f2 << "~~~~~~~~~~~~~~~~|~~~~~~~~~~~~~~~~~~~~~~~|~~~~~~~~~~~~~~~~~~~~~~~|~~~~~~~~~~~~~~~~~~~~~~~|~~~~~~~~~~~~~~~~~~~~~~~\n";
102. }
103. //Вывод в файл.
104. while (i < countStlb)
105. {
106. str = arrayIndex[i]; //Взятие индекса из массива индексов.
107. if (array[4][str] != "" && array[4][str] != "----")
108. {
109. f2 << " " << array[4][str] << tab(array[4][str]); //Вывод 1-м столбец Порта Прибытия.
110. f2 << array[0][str] << tab2(array[0][str]) << "\t\t|\t";
111. f2 << array[5][str] << tab2(array[5][str]);
112. f2 << array[6][str] << "\n\t\t|\t\t\t|\t";
113. f2 << array[1][str] << tab2(array[1][str]);
114. f2 << array[2][str] << tab2(array[2][str]);
115. f2 << array[3][str] << "\n";
116. f2 << "----------------|-----------------------|-----------------------|-----------------------|-----------------------\n";
117. }
118. i++;
119. }
120. }
121. cout << "\n Сортировка завершена.\n Файл <<" << fileName << ">> сохранён по пути: ";
122. f2.close();
123. }
124. //Сортировка по возрастанию Рейсов, Портов Отправления/Прибытия. (Сортирует заданный столбец с запоминанием перемещаемых индексов(Номера Строки))
125. void sortUp(int countStlb, int nomberStlb, string\*\* array, string fileName)
126. //countStlb – число элементов в сортируемом столбце.
127. //nomberStlb - номер сортируемого столбца
128. //Array - сортируемый массив.
129. //fileName - Имя выходного файла.
130. {
131. int i(0), j(0), k(0);
132. string\* arrayStr = new string[countStlb]; //Массив сортируемых параметров.
133. int\* arrayIndex = new int[countStlb]; //Массив индексов параметров.
134. //Запись в Локальный Массив.
135. for (i = 0; i < countStlb; i++)
136. {
137. arrayStr[i] = array[nomberStlb][i]; //Запись параметров в массив параметров.
138. arrayIndex[i] = i; //Запись параметра индекса.
139. }
140. //Сортировка Локального Массива.
141. for (i = 0; i < countStlb; i++) //Цикл проверки предыдущих значений.
142. {
143. j = i;
144. for (k = i + 1; k < countStlb; k++) //Цикл проверки следующих значений.
145. {
146. if (arrayStr[k] < arrayStr[j]) //Если следующий эл-т меньше предыдущего.
147. {
148. j = k;
149. }
150. }
151. swap(arrayStr[j], arrayStr[i]); //Меняем местами параметры
152. swap(arrayIndex[j], arrayIndex[i]); //Меняем местами индексы параметров.
153. }
154. vivod(countStlb, array, arrayIndex, fileName);
155. delete[] arrayStr; //Освобождение памяти.
156. delete[] arrayIndex; //Освобождение памяти.
157. }
158. //Сортировка по убыванию Рейсов, Портов Отправления/Прибытия. (Сортирует заданный столбец с запоминанием перемещаемых индексов(Номера Строки))
159. void sortDown(int countStlb, int nomberStlb, string\*\* array, string fileName)
160. //countStlb – число элементов в сортируемом столбце.
161. //nomberStlb - номер сортируемого столбца
162. //Array - сортируемый массив.
163. //fileName - Имя выходного файла.
164. {
165. int i(0), j(0), k(0);
166. string\* arrayStr = new string[countStlb]; //Массив сортируемых параметров.
167. int\* arrayIndex = new int[countStlb]; //Массив индексов параметров.
168. //Запись в Локальный Массив.
169. for (i = 0; i < countStlb; i++)
170. {
171. arrayStr[i] = array[nomberStlb][i]; //Запись параметров в массив параметров.
172. arrayIndex[i] = i; //Запись параметра индекса.
173. }
174. //Сортировка Локального Массива.
175. for (i = 0; i < countStlb; i++) //Цикл проверки предыдущих значений.
176. {
177. j = i;
178. for (k = i + 1; k < countStlb; k++) //Цикл проверки следующих значений.
179. {
180. if (arrayStr[k] > arrayStr[j]) //Если следующий эл-т меньше предыдущего.
181. {
182. j = k;
183. }
184. }
185. swap(arrayStr[j], arrayStr[i]); //Меняем местами параметры.
186. swap(arrayIndex[j], arrayIndex[i]); //Меняем местами индексы параметров.
187. }
188. vivod(countStlb, array, arrayIndex, fileName);
189. delete[] arrayStr; //Освобождение памяти.
190. delete[] arrayIndex; //Освобождение памяти.
191. }

MySort2xStr.h //Сортировка 2-х аргументов и вывод таблицы в файл.

1. #include "MySort.h" //Заголовочный файл функции вывода и сортировки по одному аргументу.
2. #include <vector>
3. using namespace std;
4. //Функция вывода отсортированной таблицы.
5. void vivod2(int countStlb, string\*\* array, int\* arrayIndex, string\* arrayStr, string fileName)
6. //countStlb – номер сортируемого столбца.
7. //Array - сортируемый массив.
8. //arrayIndex – Массив индексов отсортированных элементов.
9. //arrayStr - Массив отсортированных строк.
10. //fileName - Имя выходного файла.
11. {
12. int str(0), stlb(0), i(0), j(0), a(0),b(0); //Счетчик.
13. fileName = ifFileDetected(fileName); //Задать новое имя файла.
14. ofstream f2(fileName); //Открыть файл для заполнения.
15. vector<string>localVec1(countStlb); //Локальный вектор Страны/Моря Отправления.
16. vector<string>localVec2(countStlb); //Локальный вектор Страны/Моря Прибытия.
17. //Вывод в файл Морей.
18. if (!fileName.find("Sea")) //Если встречен файл Морей.
19. {
20. //Копирование из исходного массива в локальный вектор.
21. for (i = 0; i < countStlb; i++)
22. {
23. localVec1[i] = array[3][i];
24. localVec2[i] = array[6][i];
25. }
26. //Вывод в файл заголовка.
27. {
28. if (fileName.find("Up") != string::npos) //Если сортировка по возрастанию.
29. {
30. f2 << " Таблица отсортирована по параметру <<Море>> по Возрастанию.\n\n";
31. }
32. else //Если сортировка по убыванию.
33. {
34. f2 << " Таблица отсортирована по параметру <<Море>> по Убыванию.\n\n";
35. }
36. f2 << " Море:\t\t|\tРейс:\t\t|\tПорт:\t\t|\tСтрана:\n"; //Вывести 1-м столбец морей.
37. f2 << "~~~~~~~~~~~~~~~~|~~~~~~~~~~~~~~~~~~~~~~~|~~~~~~~~~~~~~~~~~~~~~~~|~~~~~~~~~~~~~~~~~~~~~~~\n";
38. }
39. //Вывод в файл эллементов.
40. while (stlb < countStlb)
41. {
42. str = arrayIndex[stlb]; //Взятие индекса из массива индексов.
43. if (arrayStr[stlb] == localVec1[str] && localVec1[str] != "----") //Если отсортированное слово встречено
44. //в массиве отправления.
45. {
46. f2 << " " << localVec1[str] << tab(localVec1[str]); //Вывести Море отправления Рейса.
47. f2 << array[0][str] << tab2(array[0][str]); //Вывести Рейс.
48. f2 << array[1][str] << tab2(array[1][str]); //Вывести Порт Отправления Рейса.
49. f2 << array[2][str] << "\n"; //Вывести Страну Отправления Рейса.
50. f2 << "----------------|-----------------------|-----------------------|-----------------------\n";
51. localVec1[str] = "0"; //"Удалить" слово из Вектора Отправления.
52. } //Фикс многократного вывода сортируемого столбца.
53. if (localVec2[str] == arrayStr[stlb] && localVec2[str] != "----")
54. {
55. f2 << " " << localVec2[str] << tab(localVec2[str]); //Вывести Море Прибытия Рейса.
56. f2 << array[0][str] << tab2(array[0][str]); //Вывести Рейс.
57. f2 << array[4][str] << tab2(array[4][str]); //Вывести Порт Прибытия Рейса.
58. f2 << array[5][str] << "\n"; //Вывести Страну Прибытия Рейса.
59. f2 << "----------------|-----------------------|-----------------------|-----------------------\n";
60. localVec2[str] = "0"; //"Удалить" слово из Вектора Прибытия.
61. } //Фикс многократного вывода сортируемого столбца.
62. stlb++;
63. }
64. }
65. //Вывод в файл Стран.
66. if (!fileName.find("Country")) //Если встречен файл Стран.
67. {
68. for (i = 0; i < countStlb; i++)
69. {
70. localVec1[i] = array[2][i];
71. localVec2[i] = array[5][i];
72. }
73. //Вывод в файл заголовка.
74. {
75. if (fileName.find("Up") != string::npos) //Если сортировка по возрастанию.
76. {
77. f2 << " Таблица отсортирована по параметру <<Страна>> по Возрастанию.\n\n";
78. }
79. else //Если сортировка по убыванию.
80. {
81. f2 << " Таблица отсортирована по параметру <<Страна>> по Убыванию.\n\n";
82. }
83. f2 << " Страна:\t|\tРейсы:\t\t|\tПорт:\t\t|\tМоре:\n"; Вывести 1-м столбец Стран.
84. f2 << "~~~~~~~~~~~~~~~~|~~~~~~~~~~~~~~~~~~~~~~~|~~~~~~~~~~~~~~~~~~~~~~~|~~~~~~~~~~~~~~~~~~~~~~~\n";
85. }
86. //Вывод в файл эллементов.
87. while (stlb < countStlb)
88. {
89. str = arrayIndex[stlb]; //Взятие индекса из массива индексов.
90. a = 0; b = 0;
91. if (arrayStr[stlb] == localVec1[str] && localVec1[str] != "----") //Если отсортированное слово встречено
92. { //в массиве отправления.
93. f2 << " " << localVec1[str] << tab(localVec1[str]); //Вывести Страну Отправления Рейса.
94. f2 << array[0][str] << tab2(array[0][str]); //Вывести Рейс.
95. f2 << array[1][str] << tab2(array[1][str]); //Вывести Порт Отправления Рейса.
96. f2 << array[3][str] << "\n"; //Вывести Море Отправления Рейса.
97. f2 << "----------------|-----------------------|-----------------------|-----------------------\n";
98. localVec1[str] = "0"; //"Удалить" слово из массива отправления.
99. }
100. if (localVec2[str] == arrayStr[stlb] && localVec2[str] != "----") //Если отсортированное слово встречено
101. { //в массиве Прибытия.
102. f2 << " " << localVec2[str] << tab(localVec2[str]); //Вывести Страну Прибытия Рейса.
103. f2 << array[0][str] << tab2(array[0][str]); //Вывести Рейс.
104. f2 << array[4][str] << tab2(array[4][str]); //Вывести Порт Прибытия Рейса.
105. f2 << array[6][str] << "\n"; //Вывести Море Прибытия Рейса.
106. f2 << "----------------|-----------------------|-----------------------|-----------------------\n";
107. localVec2[str] = "0"; //"Удалить" слово из массива Прибытия.
108. }
109. stlb++;
110. }
111. }
112. cout << "\n Сортировка завершена.\n Файл <<" << fileName << ">> сохранён по пути: ";
113. f2.close();
114. }
115. //Сортировка по возрастанию Двух Морей или Стран. (Сортирует заданный столбец с запоминанием перемещаемых индексов(Номера Строки))
116. void sortUp2(int countStlb, int nomberStlb, string\*\* array, string fileName)
117. //countStlb – число элементов в сортируемом столбце.
118. //nomberStlb - номер сортируемого столбца
119. //Array - сортируемый массив.
120. //fileName - Имя выходного файла.
121. {
122. int i(0), j(0), k(0), j1(0); //Счетчик.
123. string\* arrayStr = new string[countStlb]; //Массив сортируемых параметров.
124. int\* arrayIndex = new int[countStlb]; //Массив индексов параметров.
125. //Запись в Локальный Массив.
126. while (i < countStlb)
127. {
128. arrayStr[i] = array[nomberStlb][j]; //Запись 1 параметра моря или страны в массив.
129. arrayIndex[i] = j; //Запись индекса 1-го параметра.
130. string strr("");
131. if (array[nomberStlb + 3][j] != strr)
132. {
133. arrayStr[i + 1] = array[nomberStlb + 3][j1]; //Запись 2-го параметра моря или страны в массив.
134. arrayIndex[i + 1] = j; //Запись индекса 2-го параметра в массив.
135. i++; j1++;
136. }
137. i++; j++;
138. }
139. //Сортировка в Локального Массива.
140. for (i = 0; i < countStlb; i++) //Цикл проверки предыдущих значений.
141. {
142. j = i;
143. for (k = i + 1; k < countStlb; k++) //Цикл проверки следующих значений.
144. {
145. if (arrayStr[k] < arrayStr[j]) //Если следующий эл-т меньше предыдущего.
146. {
147. j = k;
148. }
149. }
150. swap(arrayStr[j], arrayStr[i]); //Меняем местами параметры.
151. swap(arrayIndex[j], arrayIndex[i]); //Меняем местами индексы параметров.
152. }
153. vivod2(countStlb, array, arrayIndex, arrayStr, fileName);
154. delete[] arrayStr; //Освобождение памяти.
155. delete[] arrayIndex; //Освобождение памяти.
156. }
157. //Сортировка по убыванию Двух Морей или Стран. (Сортирует заданный столбец с запоминанием перемещаемых индексов(Номера Строки))
158. void sortDown2(int countStlb, int nomberStlb, string\*\* array, string fileName)
159. //countStlb – число элементов в сортируемом столбце.
160. //nomberStlb - номер сортируемого столбца
161. //Array - сортируемый массив.
162. //fileName - Имя выходного файла.
163. {
164. int i(0), j(0), k(0), j1(0); //Счетчик.
165. string\* arrayStr = new string[countStlb]; //Массив сортируемых параметров.
166. int\* arrayIndex = new int[countStlb]; //Массив индексов параметров.
167. //Запись в Локальный Массив.
168. while (i < countStlb)
169. {
170. arrayStr[i] = array[nomberStlb][j]; //Запись 1 параметра моря или страны в массив.
171. arrayIndex[i] = j; //Запись индекса 1-го параметра.
172. string strr("");
173. if (array[nomberStlb + 3][j] != strr)
174. {
175. arrayStr[i + 1] = array[nomberStlb + 3][j1]; //Запись 2-го параметра моря или страны в массив.
176. arrayIndex[i + 1] = j; //Запись индекса 2-го параметра в массив.
177. i++; j1++;
178. }
179. i++; j++;
180. }
181. //Сортировка Локального Массива.
182. for (i = 0; i < countStlb; i++) //Цикл проверки предыдущих значений.
183. {
184. j = i;
185. for (k = i + 1; k < countStlb; k++) //Цикл проверки следующих значений.
186. {
187. if (arrayStr[k] > arrayStr[j]) //Если следующий эл-т меньше предыдущего.
188. {
189. j = k;
190. }
191. }
192. swap(arrayStr[j], arrayStr[i]); //Меняем местами параметры.
193. swap(arrayIndex[j], arrayIndex[i]); //Меняем местами индексы параметров.
194. }
195. vivod2(countStlb, array, arrayIndex, arrayStr, fileName);
196. delete[] arrayStr; //Освобождение памяти.
197. delete[] arrayIndex; //Освобождение памяти.
198. }

Loading.h //Функция печатающая точки с заданным интервалом.

1. #include <iostream>
2. #include <Windows.h>
3. using namespace std;
4. void Loading(int MiliSek,int Kolich\_Tochek) //Функция печатающая точки с заданным интервалом.
5. //MiliSek – кол-во миллисекунд задержки.
6. //Kolich\_Tochek – колличество печатаемых точек.
7. {
8. int i = 0;
9. do
10. {
11. i++;
12. cout << ". ";
13. Sleep(MiliSek);
14. } while (i != Kolich\_Tochek);
15. }

Start.h // Анимация инициализации программы.

1. #include <conio.h>
2. #include "Loading.h"
3. using namespace std;
4. void Start()
5. { //Функция Анимация загрузки.
6. cout << " Запустить программу(Enter)\n";
7. \_getch();
8. cout << " Запуск программы:\n";
9. Loading(60, 10);
10. for (int i = 0; i < 2; i++)
11. {
12. system("cls");
13. cout << " Запустить программу(Enter)\n";
14. cout << " Запуск программы:\n";
15. Loading(60, 10);
16. Sleep(100);
17. }
18. }

Tabulation on Tabl.h //Функции табулирования в файлах.

1. string tab(string word) //Функция табуляции 1-го столбца.
2. //Word – слово, относительного которого табулируется таблица.
3. {
4. if (word.size() > strlen(" ")) //Корректировка вывода по столбцу Портов.
5. {
6. return ("\t|\t");
7. }
8. else
9. {
10. return ("\t\t|\t");
11. }
12. }
13. string tab2(string word) //Функция табуляции 2+ столбцов.
14. //Word – слово, относительного которого табулируется таблица.
15. {
16. if (word.size() > strlen(" ")) //Корректировка вывода по столбцу Портов.
17. {
18. return ("\t|\t");
19. }
20. else
21. {
22. return ("\t\t|\t");
23. }
24. }
25. string tab3(string word) //Функция табуляции 2+ столбцов.
26. //Word – слово, относительного которого табулируется таблица.
27. {
28. if (word.size() < strlen(" ")) //Корректировка вывода по столбцу Портов.
29. {
30. return ("\t\t\t|\t");
31. }
32. else
33. {
34. if (word.size() < strlen(" "))
35. {
36. return ("\t\t|\t");
37. }
38. else
39. return ("\t|\t");
40. }
41. }

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

ifFile.h //Функция увеличения номера файла в его названии.

* 1. string ifFileDetected(string oldFileName) //oldFileName - Имя файла.
  2. {
  3. int pos1(0);
  4. int pos2(0);
  5. int y(0);
  6. string str;
  7. while (1 > 0)
  8. {
  9. ifstream f4(oldFileName);
  10. if (f4.is\_open())
  11. {
  12. pos1 = oldFileName.find("(", 0) + 1; //Первая позиция номера файла.
  13. pos2 = oldFileName.find(")", 0); //Последняя позиция номера файла.
  14. str = oldFileName.substr(pos1, pos2 - pos1); //Считать номер файла.
  15. y = stoi(str.c\_str()) + 1; //Увеличить номер на 1.
  16. str = to\_string(y);
  17. oldFileName.replace(pos1, pos2 - pos1, str); //Заменить старый номер файла на новый.
  18. }
  19. else
  20. {
  21. return oldFileName; //Вернуть в функцию новое имя файла.
  22. }
  23. f4.close();
  24. }
  25. }