«Московский авиационный институт»   
(Национальный исследовательский университет)

Лабораторная работа по курсу

«Технологии программирования»

Часть 1

Вариант 24

Выполнил:   
студент группы

 М7О-207С-19

Понкращенков Дмитрий

Проверил:

Барчев Н.Б.

Москва

**Содержание**

1. Задание. ……………………………………………………………………………………………………………..3
2. Псевдокод. ………………………………………………………………………………………………………...4
3. Сведения о программной реализации. …………………………………………………………...7
4. Инструкция пользователя. ………………………………………………………………………………..9
5. Листинг. …………………………………………………………………………………………………………...11
6. **Задание**   
   1. Подготовить программу, формирующую на основе информации, вводимой пользователем с клавиатуры, два внешних файла:
   2. Файл портов: состоит из записей, каждая из которых включает три поля - название порта, страны, моря;
   3. Файл рейсов: состоит из записей, каждая из которых включает три поля - названия судна, порта отправления и прибытия.
   4. Подготовить вторую программу, формирующие сведенья о перевозках, осуществляемых судами, с указанием стран и морей на основе данных из двух внешних файлов, сформированных в результате работы программы, подготовленной в п.1.
   5. В процессе проектирования предусмотреть необходимые по смыслу задания проверки корректности данных, а также адекватное задаче взаимодействие  с пользователем.   
      Во второй программе реализовать возможность вывода итоговой информации в двух режимах: избирательно на экран (по запросу пользователем конкретных данных с клавиатуры) и полностью - в отдельный текстовый файл. Оба режима вывода должны предоставлять сгруппированную по смыслу информацию из обоих входных файлов в виде, удобном для восприятия пользователем.   
      Учесть, что ограничения на размеры файлов отсутствуют, файлы в общем случае могут быть различной длины.   
      В обязательном порядке использовать языковые средства организации программных единиц.   
      Не использовать программные средства, поддерживающие работу с базами данных.
   6. Программы подготовить и использованием средств одной из реализаций языка программирования С++.
7. **Псевдокод.**

Объявление функции поиска файла.

Если файл открыт, то конец функции.

Объявление функции проверки корректности ввода.

Задается строка с запрещенными символами.

Пока запрещенные символы не найдены, проверять строку на их наличие.

Если символы найдены, предупредить пользователя об этом и запросить новый ввод строки.

Объявление функции поиска пустого ввода в строку.

Если длина строки равна 0 или введённый символ пробел,

То вывести “------“

Объявление функции замены пробела на тире

Если Пробел найден заменить его на тире.

Создание главного меню программы:

* Редактировать файл Портов (1)
* Редактировать файл Рейсов (2)
* Выйти из программы. (esc)
* Если выбран 1 вариант:

Проверить наличие файла портов

* Если файл существует
* Удалить файл или создать новый. (BackSpace)
* Редактировать файл. (Enter)
* Выйти в меню программы. (esc)
* Если выбран вариант создания нового файла:

Открыть файл для редактирования с нуля.

* Если выбран выход в меню:

Показать анимацию загрузки

Очистить экран

Выйти в меню программы

* Если файл открыт для редактирования:
* Запросить название порта

Если название порта не было введено запросить его еще раз.

Проверить название порта на: Запрещенные символы.

Наличие пробелов.

Заменить первую букву на заглавную.

Записать строку в файл.

* Запросить название страны

Проверить название страны на: Запрещенные символы.

Наличие пробелов.

Пустой ввод.

Заменить первую букву на заглавную.

Записать строку в файл.

* Запросить название моря

Проверить название моря на: Запрещенные символы.

Наличие пробелов.

Пустой ввод.

Заменить первую букву на заглавную.

Записать строку в файл.

* Анимация сохранения файла.
* Предложить выйти в главное меню программы или продолжить ввод данных.
* Если файла не существовало.

Создать файл и перейти к его редактированию.

* Если выбран 2 вариант:

Проверить наличие файла рейсов.

* Если файл существует.
  + Удалить файл или создать новый. (BackSpace)
  + Редактировать файл. (Enter)
  + Выйти в меню программы. (esc)
* Если выбран вариант создания нового файла:

Открыть файл для редактирования с нуля.

* Если выбран выход в меню:

Показать анимацию загрузки.

Очистить экран.

Выйти в меню программы.

* Если файл открыт для редактирования:
* Запросить название рейса

Если название рейса не было введено запросить его еще раз.

Проверить название рейса на: Запрещенные символы.

Наличие пробелов.

Заменить первую букву на заглавную.

Записать строку в файл.

* Запросить название Порта отправления

Проверить название на: Запрещенные символы.

Наличие пробелов.

Пустой ввод.

Заменить первую букву на заглавную.

Записать строку в файл.

* Запросить название Порта прибытия

Проверить название на: Запрещенные символы.

Наличие пробелов.

Пустой ввод.

Заменить первую букву на заглавную.

Записать строку в файл.

* Анимация сохранения файла.
* Предложить выйти в главное меню программы или продолжить ввод данных.
* Если файла не существовало.

Создать файл и перейти к его редактированию.

* Если выбран 3 вариант:

Проиграть анимацию загрузки

Завершить программу.

* Если была введена неправильная клавиша:

Предупредить пользователя

Вернуться к главному меню.

1. **Сведения о программной реализации:**

Версия системы программирования:   
Microsoft Visual Studio Community 2019 Version 16.3.8   
  
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 | * 2 ГБ ОЗУ, рекомендуется 4 ГБ ОЗУ |

**Описание входных и выходных данных:**

Входными данными для следующих файлов будут вводимые пользователем с клавиатуры:

1. Для файла портов:
   * + Название порта;
     + Название страны;
     + Название моря;
2. Для файла рейсов:

* Название рейса;
* Порт отправления;
* Порт прибытия;

Выходными данными будут два текстовых файла: файл Портов с названием Port.txt и файл Рейсов с названием Rase.txt, содержащие данные, описанные выше.  Файлы будут автоматически созданы в одной папке с программой.

**Особые указания по вводу параметров:**

* Параметры Название Рейса и Название Порта не могут быть пустыми.
* Никакие параметры не могут содержать цифры или служебные символы кроме тире (-) и пробела.

|  |  |
| --- | --- |
| Диагностические сообщения | Причина возникновения |
| «Введена неправильная клавиша» | При выборе варианта была нажата неверная клавиша, т.е. была нажата любая клавиша, кроме «1», «2», «Esc» |
| Возможно, Вы ввели цифры или некорректные символы: []{}+=-&^:;|/><$#@.,()!?№% | При вводе данных в структуры были использованы неподходящие символы |

**Программные единицы:**

Функция: **isfile** (*string filepatch*).

Назначение: поиск файла.

Тип: bool;

Входные параметры: filepatch путь к файлу.

Тип: string;

Возвращаемое значение checks;

Тип: bool;

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

Функция **Checking** (*string field, string s*).

Назначение: проверка корректности ввода.

Тип: string;

Входные параметры: «S» - путь к файлу, Тип: string;

«Field» - проверяемая строка, Тип: string;

Возвращаемое значение checks;

Тип: bool;

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

Функция **None** (*string field*)

Назначение: Замена пустого ввода на (-----).

Тип: string;

Входные параметры: «Field» - проверяемая строка, Тип: string;

Возвращаемое значение field;

Тип: string;

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

Функция **Tire** (*string field*)

Назначение: Замена вводимого пробела на Тире (-).

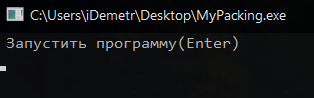
Входные параметры: «Field» - проверяемая строка, Тип: string;

Возвращаемое значение field;

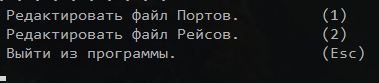
Тип: string;

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

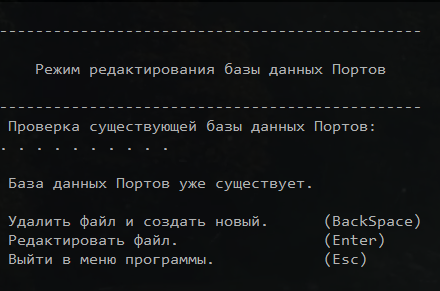
1. **Инструкция пользователя:**

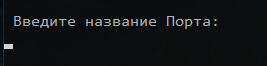
После запуска программы на экране монитора появляется сообщение с предложением загрузить программу:

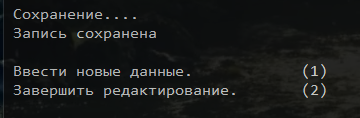
Для продолжения программы необходимо нажать клавишу «Enter».

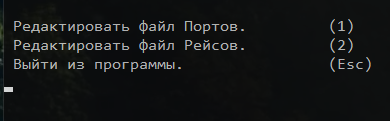
После проигрывания анимации загрузки программы на экране монитора появляется меню с выбором действий:

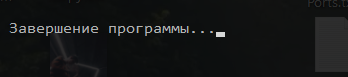
Выбор пунктов осуществляется с помощью ввода нужного номера пункта с клавиатуры.

После выбора пользователем, например, первого пункта меню, если файл был обнаружен, на экране появится следующее сообщение:

После можно будет начать запись в файл:

После записи всех данных в файл на экране программы появится следующее окно:

При выборе Завершения редактирования (2) будет произведена очистка экрана и переход в главное меню:

При выборе варианта №3 «Выйти из программы (Esc)» будет проигранна анимация завершения программы и последующий выход из нее:

1. **Листинг.**

#include <iostream>

#include <fstream>

#include <string>

#include <conio.h>

#include <Windows.h>

#include <string.h>

#include <stdio.h>

using namespace std;

bool isfile(std::string filepath) //Функция проверки наличия файла.

{

bool check = false;

std::ifstream fin(filepath.c\_str());

if (fin.is\_open())

check = true;

fin.close();

return check;

}

string Cheking(std::string field, std::string s) //Функция для проверки корректности ввода.

{

bool checks = true;

string Symbols = "[]{}+=&^:;|/><$#@.,!?№%()1234567890";

while (checks == true)

{

unsigned int k = 0;

for (unsigned int i = 0; i < field.length(); i++)

{

for (unsigned int i1 = 0; i1 < Symbols.length(); i1++)

{

if (field[i] == Symbols[i1])

{

k = k + 1;

}

}

}

if (k >= 1 || k == field.length() && field.length() != 0)

{

cout << s << " Возможно, Вы ввели цифры или некорректные символы: ( []{}+=-&^:;|/><$#@.,()!?№% )\n Попробуйте ещё раз: \n";

getline(cin, field);

}

else checks = false;

}

checks = true;

return field;

}

string None(std::string field) //Функция если ввод отсутсвует.

{

if (field.length() == 0 || field == " ")

field = " ---- ";

return field;

}

string Tire(std::string field) //Функция замены пробелов на Тире(-).

{

for (unsigned int r = 0; r < field.length() + 1; r++)

{

if (field[r] == ' ')

field[r] = '-';

};

return field;

};

struct SPort

{

string NamePort;

string Contry;

string Sea;

};

struct SRase

{

string NameSudna;

string PortOtpr;

string PortPribit;

};

int main()

{

setlocale(LC\_ALL, "Russian");

SetConsoleCP(1251); //Русский язык в консоли.

SetConsoleOutputCP(1251); //Русский язык в файле.

cin.sync();

char ch = 0;

int i = 0;

int x = 0;

bool flag = false;

string PortsName = "Ports.txt"; //Название файла.

string RaseName = "Rase.txt";

SPort FPort;

SRase FRase;

cout << " Запустить программу(Enter)\n";

\_getch() == 13;

cout << " Запуск программы:\n";

do

{ //Анимация загрузки.

i++;

cout << ". ";

Sleep(200);

} while (i != 10);

while (!flag) //Чекпоинт.

{

cout << "\n Редактировать файл Портов. (1)\n";

cout << " Редактировать файл Рейсов. (2)\n";

cout << " Выйти из программы. (Esc)\n";

ch = \_getch();

if (ch == 49)

{ //Редактирование факла Портов.

system("cls");

cout << "\n-----------------------------------------------\n";

cout << "\n Режим редактирования базы данных Портов\n";

cout << "\n-----------------------------------------------\n";

cout << " Проверка существующей базы данных Портов:\n";

i = 0;

do

{ //Анимация загрузки.

i++;

cout << ". ";

Sleep(200);

} while (i != 10);

if (isfile(PortsName)) //Проверка наличия файла.

{

while (x < 10)

{

cout << "\n\n База данных Портов уже существует.\n\n";

cout << " Удалить файл и создать новый. (BackSpace)\n";

cout << " Редактировать файл. (Enter)\n";

cout << " Выйти в меню программы. (Esc)\n";

ch = \_getch();

if (ch == 8)

{ //Создаём чистый файл

cout << "\n База данных обновленна.\n" << ch;

ofstream port(PortsName);

flag = true;

break;

}

else

if (ch == 13)

{

flag = true;

break;

}

else

if (ch == 27) //Переход в меню.

{

cout << " Загрузка меню";

i = 0;

do

{ //Анимация загрузки

i++;

cout << ".";

Sleep(300);

} while (i != 4);

system("cls");

x = 10;

flag = false;

}

else cout << "\n Введена неправильная клавиша\n";

}

}

else

{

cout << "\n Базы данных Портов не существует и будет создана\n";

flag = true;

}

while (flag == true) //Чекпоинт.

{

ofstream Port(PortsName, ofstream::app);

x = 0;

string s1 = " [Порт] ";

string s2 = " [Страна] ";

string s3 = " [Море] ";

while (x < 10)

{

cout << "\n Введите название Порта: \n";

getline(cin, FPort.NamePort);

while (flag == true)

{

if (FPort.NamePort.length() == 0)

{

cout << "\n Название Судна должно быть обязательно заполнено\n";

cout << "\n Введите название Судно заново\n";

getline(cin, FPort.NamePort);

}

else

flag = false;

}

flag = true;

FPort.NamePort = Cheking(FPort.NamePort, s1);

FPort.NamePort = Tire(FPort.NamePort);

FPort.NamePort[0] = toupper(FPort.NamePort[0]);

if (FPort.NamePort.length() >= strlen(" "))

Port << FPort.NamePort << "\t\t";

else

Port << FPort.NamePort << "\t\t\t";

cout << " Введите название Страны: \n";

getline(cin, FPort.Contry);

FPort.Contry = Cheking(FPort.Contry, s2);

FPort.Contry = Tire(FPort.Contry);

FPort.Contry = None(FPort.Contry);

FPort.Contry[0] = toupper(FPort.Contry[0]);

if (FPort.Contry.length() >= strlen(" "))

Port << FPort.Contry << "\t\t";

else

Port << FPort.Contry << "\t\t\t";

cout << " Введите название Моря: \n";

getline(cin, FPort.Sea);

FPort.Sea = Cheking(FPort.Sea, s3);

FPort.Sea = Tire(FPort.Sea);

FPort.Sea = None(FPort.Sea);

FPort.Sea[0] = toupper(FPort.Sea[0]);

Port << FPort.Sea << "\n";

cout << "\n Сохранение";

i = 0;

do

{ //Анимация загрузки

i++;

cout << ".";

Sleep(300);

} while (i != 4);

cout << "\n Запись сохранена\n\n Ввести новые данные. (1)\n Завершить редактирование. (2)";

ch = \_getch();

if (ch == 50)

{

system("cls");

break;

}

}

Port.close();

flag = false;

}

}

else

if (ch == 50)

{ //Редактирование файла Рейсов

system("cls");

cout << "\n-----------------------------------------\n";

cout << " Режим редактирования базы данных Рейсов\n";

cout << "\n-----------------------------------------\n";

cout << " Проверка существующей базы данных Рейсов:\n";

i = 0;

do

{ //Анимация загрузки

i++;

cout << ". ";

Sleep(200);

} while (i != 10);

if (isfile(RaseName))

{

while (x < 10)

{

cout << "\n\n База данных Рейсов уже существует.\n\n";

cout << " Удалить файл и создать новый. (BackSpace)\n";

cout << " Редактировать файл. (Enter)\n";

cout << " Выйти в меню программы. (Esc)\n";

ch = \_getch();

if (ch == 8)

{ //Создаём чистый файл Рейсов

cout << "\n База данных обновленна.\n" << ch;

ofstream Rase(RaseName);

flag = true;

break;

}

else

if (ch == 13)

{

flag = true;

break;

}

else

if (ch == 27) //Переход в меню.

{

cout << " Загрузка меню";

i = 0;

do

{ //Анимация загрузки

i++;

cout << ".";

Sleep(300);

} while (i != 4);

system("cls");

x = 10;

flag = false;

}

else cout << "\n Введена неправильная клавиша\n";

}

}

else

{

cout << "\n Базы данных Портов не существует и будет создана\n";

flag = true;

}

while (flag == true) //Чекпоинт.

{

ofstream Rase(RaseName, ofstream::app);

x = 0;

string s1 = " [Судно] ";

string s2 = " [Порт отправления] ";

string s3 = " [Порт прибытия] ";

while (x < 10)

{

cout << "\n Введите Название Судна: \n";

getline(cin, FRase.NameSudna);

while (flag == true)

{

if (FRase.NameSudna.length() == 0)

{

cout << "\n Название Судна должно быть обязательно заполнено\n";

cout << "\n Введите название Судно заново\n";

getline(cin, FRase.NameSudna);

}

else

flag = false;

}

flag = true;

FRase.NameSudna = Cheking(FRase.NameSudna, s1);

FRase.NameSudna = Tire(FRase.NameSudna);

FRase.NameSudna[0] = toupper(FRase.NameSudna[0]);

if (FRase.NameSudna.length() >= strlen(" "))

Rase << FRase.NameSudna << "\t\t";

else

Rase << FRase.NameSudna << "\t\t\t";

cout << "\n Введите название Порта отправления: \n";

getline(cin, FRase.PortOtpr);

FRase.PortOtpr = Cheking(FRase.PortOtpr, s2);

FRase.PortOtpr = Tire(FRase.PortOtpr);

FRase.PortOtpr = None(FRase.PortOtpr);

FRase.PortOtpr[0] = toupper(FRase.PortOtpr[0]);

if (FRase.PortOtpr.length() >= strlen(" "))

Rase << FRase.PortOtpr << "\t\t";

else

Rase << FRase.PortOtpr << "\t\t\t";

cout << " Введите название Порты прибытия: \n";

getline(cin, FRase.PortPribit);

FRase.PortPribit = Cheking(FRase.PortPribit, s3);

FRase.PortPribit = Tire(FRase.PortPribit);

FRase.PortPribit = None(FRase.PortPribit);

FRase.PortPribit[0] = toupper(FRase.PortPribit[0]);

Rase << FRase.PortPribit << "\n";

cout << "\n Сохранение";

i = 0;

do

{ //Анимация загрузки

i++;

cout << ".";

Sleep(300);

} while (i != 4);

cout << "\n Запись сохранена\n\n Ввести новые данные. (1)\n Завершить редактирование. (2)\n";

ch = \_getch();

if (ch == 50)

{

system("cls");

break;

}

}

Rase.close();

flag = false;

}

}

else

if (ch == 27)

{

cout << "\n Завершение программы";

i = 0;

do

{ //Анимация загрузки.

i++;

cout << ".";

Sleep(500);

} while (i != 4);

return 0;

}

else cout << "\n Введена неправильная клавиша\n";

}

}