00ФЕДЕРАЛЬНОЕ АГЕНТСТВО ПО ОБРАЗОВАНИЮ ГОУ ВПО НИЖЕГОРОДСКИЙ ГОСУДАРСТВЕННЫЙ ТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ им. Р. Е. АЛЕКСЕЕВА

ИНСТИТУТ РАДИОЭЛЕКТРОНИКИ И ИНФОРМАЦИОННЫХ ТЕХНОЛОГИЙ

**Кафедра "Вычислительные системы и технологии"**

**ПРОГРАММИРОВАНИЕ**

**Отчёт**

**по лабораторной работе № 3**

Выполнил студент группы 23-ИВТ-2

Рябов Кирилл Александрович

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_«\_\_\_» \_\_\_\_\_\_\_ 2024 г.

Провел ст. преподаватель кафедры “ВСТ”

Мартынов Дмитрий Сергеевич

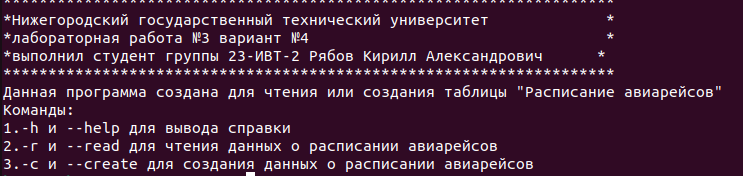
\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_«\_\_\_» \_\_\_\_\_\_\_ 2024 г.

Г. Нижний Новгород 2024

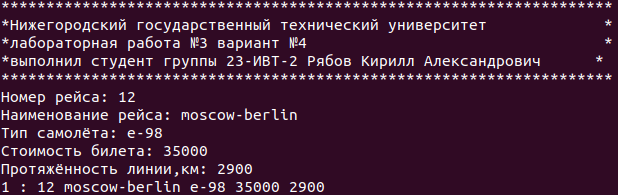
4 вариант.

Поддерживаемые опции запуска:

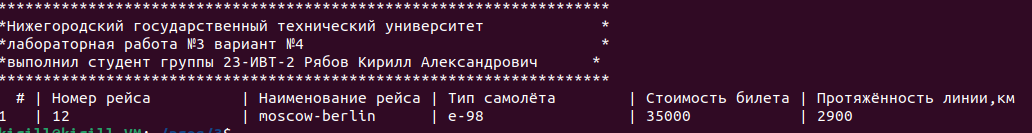
-h или --help



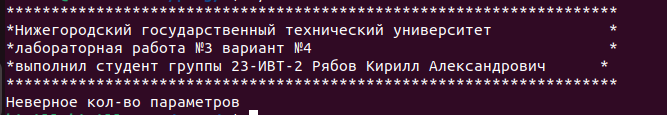
-c или --create



-r или --read



Если ничего не вводить



**Файлы:**

**main.cpp**

#include <iostream>

#include <limits>

#include <fstream>

#include <string>

#include <stdio.h>

#include <stdlib.h>

#include <iomanip>

#include <ctype.h>

#include "myLib.h"

using namespace std;

int main(int argc,char\* argv[]){

char logo[]="\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\n"

"\*Нижегородский государственный технический университет \*\n"

"\*лабораторная работа №3 вариант №4 \*\n"

"\*выполнил студент группы 23-ИВТ-2 Рябов Кирилл Александрович \*\n"

"\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\n";

cout << logo;

if (argc != 2 && argc != 4){

cout << "Неверное кол-во параметров\n";

return 0;

}

if (((argc == 2) && (string(argv[1]) != "-h")) && ((argc == 2) && (argv[1] != "--help"))){

cout << "Без файла только для функциий -h и --help\n";

return 0;

}

string commands[6] = {"-c", "--create",

"-r", "--read",

"-h", "--help"}

string\* ptr = commands;

switch (searchComands(string(argv[1]), ptr)){

case 0:

if(!atoi(argv[2]) || atoi(argv[2]) <= 0 ){

cout << "Неверный параметр кол-ва строк\n";

return 0;

}

createFile(argv[2] , argv[3]);

break;

case 1:

if(!atoi(argv[2]) || atoi(argv[2]) <= 0 ){

cout << "Неверный параметр кол-ва строк\n";

return 0;

}

readFile(argv[2] , argv[3]);

break;

case 2:

printHelp();

break;

default:

printCommands();

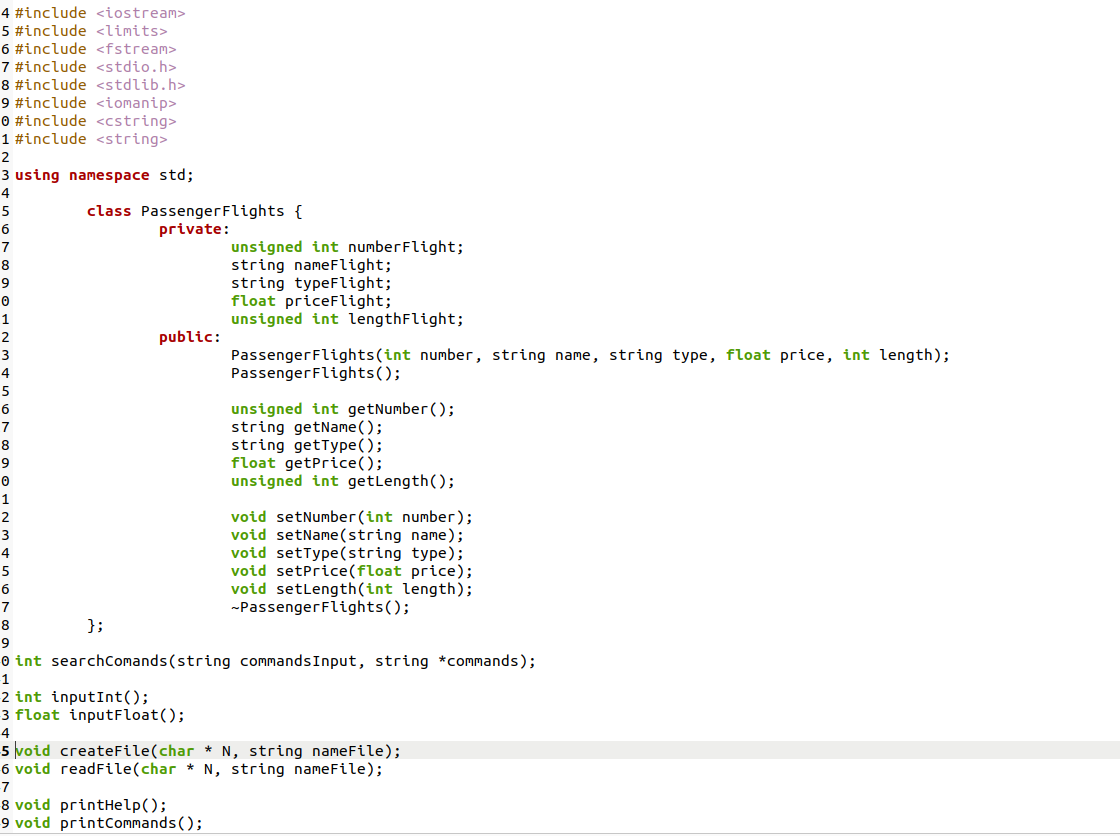
break;

}

return 0;

}

**myLib.h**



**myLib.cpp**

#include <iostream>

#include <limits>

#include <fstream>

#include <string>

#include <iomanip>

#include <cstring>

#include <string>

#include "myLib.h"

using namespace std;

PassengerFlights::PassengerFlights(int number, string name, string type, float price, int length){

numberFlight = number;

nameFlight = name;

typeFlight = type;

priceFlight = price;

lengthFlight = length;

};

PassengerFlights::PassengerFlights(){

numberFlight = 0;

nameFlight = " ";

typeFlight = " ";

priceFlight = 0;

lengthFlight = 0;

}

PassengerFlights::~PassengerFlights(){};

unsigned int PassengerFlights::getNumber(){

return numberFlight;

}

string PassengerFlights::getName(){

return nameFlight;

}

string PassengerFlights::getType(){

return typeFlight;

}

float PassengerFlights::getPrice(){

return priceFlight;

}

unsigned int PassengerFlights::getLength(){

return lengthFlight;

}

void PassengerFlights::setNumber(int number){

numberFlight = number;

}

void PassengerFlights::setName(string name){

nameFlight = name;

}

void PassengerFlights::setType(string type){

typeFlight = type;

}

void PassengerFlights::setPrice(float price){

priceFlight = price;

}

void PassengerFlights::setLength(int length){

lengthFlight = length;

}

int searchComands(string commandsInput, string \*commands){

for (int i{}; i<6; i++){

if (\*(commands+i) == commandsInput){

return i/2;

}

}

return -1;

}

int inputInt(){

int val; cin >> val;

if (!cin.good()){

throw "Ошибка";

}

if (val<0){

throw "Значения должны быть положительными";

}

cin.seekg(0, ios::end);

cin.clear();

cin.ignore(numeric\_limits<streamsize>::max(), '\n');

return val;

}

float inputFloat(){

float val; cin >> val;

if (!cin.good()){

throw "Ошибка";

}

if (val<0){

throw "Значения должны быть положительными";

}

cin.clear();

cin.ignore(numeric\_limits<streamsize>::max(), '\n');

return val;

}

void createFile(char \* N, string nameFile){

ofstream file;

file.open(nameFile);

try{

if (!file.is\_open()) throw "Ошибка открытия файла\n";

int f = atoi(N);

PassengerFlights Flights[f];

string \_;

for (int i{}; i<f; i++){

cout << "Номер рейса: ";

Flights[i].setNumber(inputInt());

cout << "Наименование рейса: ";

cin >> \_; Flights[i].setName(\_.data());

while(true)

{

for(int i = 0; i < \_.length();++i)

{

if(!isalpha(\_[i]))

{

cout<<"ERROR"<<endl;

exit (0);

cout << "Наименование рейса: ";

cin >> \_;

}

break;

}

break;

}

cout << "Тип самолёта: ";

cin >> \_; Flights[i].setType(\_.data());

cout << "Стоимость билета: ";

Flights[i].setPrice(inputFloat());

cout << "Протяжённость линии,км: ";

Flights[i].setLength(inputInt());

cout << i+1 << " : " << Flights[i].getNumber() << ' ' << Flights[i].getName() << ' ' << Flights[i].getType() << ' ' << Flights[i].getPrice() << ' ' << Flights[i].getLength() << endl;

file << Flights[i].getNumber() << ' ';

file << Flights[i].getName() << ' ';

file << Flights[i].getType() << ' ';

file << Flights[i].getPrice() << ' ';

file << Flights[i].getLength();

file << endl;

}

}

catch(const char\* error\_message){

cout << error\_message << endl;

}

file.close();

}

void readFile(char \* N, string nameFile){

try{

fstream file;

file.open(nameFile);

if (!file.is\_open()) throw "Файл не был открыт, возможно неправильное имя\n";

int f = atoi(N), ind = 0, k = 0, i = 0;

string row, words[5];

PassengerFlights Flights[f];

cout << setw(3) << "#" << " | " ;

cout << left << setw(30) << "Номер рейса" << " | " << setw(30) << "Наименование рейса" << " | " << setw(30) << "Тип самолёта" << " | " << setw(30) << "Стоимость билета" << " | " << setw(30) << "Протяженность линии,км" << endl;

while(getline(file, row) && i < f){

while (ind < row.length()){

while (row[ind] != ' '){

words[k] += row[ind];

ind++;

if(ind > row.length()) break;

}

k++;

ind++;

}

ind = 0, k = 0;

Flights[i].setNumber(stoi(words[0]));words[0]="";

Flights[i].setName(words[1]);words[1]="";

Flights[i].setType(words[2]);words[2]="";

Flights[i].setPrice(stof(words[3]));words[3]="";

Flights[i].setLength(stoi(words[4]));words[4]="";

cout << setw(3) << left << i+1 << " | ";

cout << setw(20) << left << Flights[i].getNumber() << " | ";

cout << setw(18) << left << Flights[i].getName() << " | ";

cout << setw(19) << left << Flights[i].getType() << " | ";

cout << setw(16) << left << Flights[i].getPrice() << " | ";

cout << setw(50) << left << Flights[i].getLength() << endl;

i++;

}

if (!i) throw "Файл пуст!";

}

catch(const char\* error\_message){

cout << error\_message << endl;

}

}

void printHelp(){

cout << "Данная программа создана для чтения или создания таблицы \"Расписания пассажирских авиарейсов\"\n Команды:\n1.-h и --help для вывода справки \n2.-r и --read для чтения данных о расписании авиарейсов\n3.-c и --create для создания данных о расписании авиарейсов"<<endl;

}

void printCommands(){

cout << "Команды:\n1.-h и --help для вывода справки \n2.-r и --read для чтения данных о пасписании авиарейсов\n3.-c и --create для создания данных о расписании авиарейсов" << endl ;

cout << endl;

}

Псевдокод:

Дано Размер массива N и имя файла 3.txt, в котором будут сохранены введенные данные.

Начало

| Создаем функцию поиска команды

| | Для каждого i от 0 до 5

| | | Если (commands+i) == (commandsInput)

| | | | Вернуть i/2

| | | Иначе вернуть -1

| | Всё-Если

| Всё-Цикл

| Функция InputInt и InputFloat

| Целочисленная переменная Val

| Если Ввод не является числом

| | Вывод “Ошибка”

| Если Val < 0

| | Вывод “Значения должны быть положительными”

| Очистить буфер ввода

| Вернуть Val

| Функция createFile

| | Открыть файл для записи

| | | Если файл не открыт

| | | | Вывод “Ошибка открытия файла”

| | | Все-Если

| | Создание массива Flights размером f

| | Для каждого i от 0 до f

| | Запрашиваем ввод следующих переменных

| | | Номер рейса

| | | Наименование рейса

| | | Тип самолёта

| | | Стоимость билета

| | | Протяжённость линии, км

| | | Вывод i+1 номер, наименование, тип, стоимость, протяжённость.

| | | Запись данных в файл

| | Всё-Цикл

| В случае исключения

| | Вывод “error\_massage”

| Закрыть файл

| Функция readFile

| | Открыть файл для записи

| | | Если файл не открыт

| | | | Вывод “Файл не был открыт, возможно неправильное имя”

| | | Все-Если

| | | Создание массива Flights размером f

| | | Вывод заголовка таблицы

| | | Пока i < f

| | | | Разбить строку на слова

| | | | Установить значения для Flights [f]

| | | | Вывести данные из Flights [f]

| | | Увеличить i

| | Если i равно 0

| | | Вывод “Файл пуст”

| | В случае исключения

| | | Вывод “error\_message”

| | Все-Циклы

| Закрыть файл