|  |  |
| --- | --- |
| **Gerb-BMSTU_01** | **Министерство науки и высшего образования Российской Федерации**  Калужский филиал  федерального государственного бюджетного  образовательного учреждения высшего образования  ***«Московский государственный технический университет имени Н.Э. Баумана (национальный исследовательский университет)»***  ***(КФ МГТУ им. Н.Э. Баумана)*** |

**ФАКУЛЬТЕТ** \_***ИУК «Информатика и управление»*\_\_**\_\_\_\_\_\_\_\_\_\_\_\_

**КАФЕДРА** \_\_***ИУК4 «Программное обеспечение ЭВМ, информационные технологии»***

**ЛАБОРАТОРНАЯ РАБОТА**

**«Обработка структур и файлов последовательного доступа»**

**ДИСЦИПЛИНА: «Высокоуровневое программирование»**

|  |  |  |
| --- | --- | --- |
| Выполнил: студент гр. ИУК4-12Б | | \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ ( Карельский М.К. )  (Подпись) (Ф.И.О.) |
| Проверил: | | \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ (\_\_\_\_Пчелинцева Н.И. )  (Подпись) (Ф.И.О.) |
| Дата сдачи (защиты):  Результаты сдачи (защиты): | | |
|  | - Балльная оценка:  - Оценка: | |
| Калуга, 2020 | | |

**Цель:** приобретение практических навыков процедурного программирования средствами языка С++.

Задачи:

1. Изучить тип данных структура.
2. Познакомиться с операциями, предназначенными для работы со структурами.
3. Познакомиться с организацией файлов последовательного доступа.
4. Научиться создавать объекты файловых потоков для связи с текстовым файлом на внешнем носителе.
5. Изучить основные программные средства для последовательной обработки файлов указанного типа.
6. Научиться создавать пользовательское меню для демонстрации заданий лабораторной работы.

**Вариант 2**

**Задача 1.1**

Описать как структуру переменную время (с полями часы, минуты, секунды). Составить и протестировать функцию:

**СЛЕД\_СЕК (t, t1, d),**

которая присваивает параметру t1 время на d секунд большее, чем время t (может происходить смена суток).

**Блок-схема:**









**Листинг:**

Problem1.cpp:

#include <iostream>

#include "Refresh.h"

using namespace std;

struct time

{

int hours;

int minutes;

int seconds;

};

void next\_sec(time t, time& t1, int d);

void error(const char field[]);

void problem1()

{

char res = '0';

while (res == '0')

{

refresh("СЛЕД\_СЕК");

time t;

cout << "Введите час: ";

cin >> t.hours;

if (t.hours >= 0 && t.hours < 24)

{

cout << "Введите минуту: ";

cin >> t.minutes;

if (t.minutes >= 0 && t.minutes < 60)

{

cout << "Введите секунду: ";

cin >> t.seconds;

if (t.seconds >= 0 && t.seconds < 60)

{

cout << "Введите, на сколько секунд нужно перевести время: ";

int offset{};

cin >> offset;

next\_sec(t, t, offset);

cout << "Переведенное время: " << t.hours << ":" << t.minutes << ":" << t.seconds << endl << endl;

cout << "Введите 0, если хотите запустить программу еще раз: ";

cin.ignore(32767, '\n');

cin.get(res);

if (res == '0')

{

cin.get(res);

if (res != '\n')

{

res = '1';

}

else

{

res = '0';

}

}

else if (res != '\n')

{

cin.ignore(32767, '\n');

}

}

else

{

error("секунды");

}

}

else

{

error("минуты");

}

}

else

{

error("часа");

}

}

}

void next\_sec(time t, time& t1, int d)

{

t1.seconds = t.seconds + d;

while (t1.seconds > 59)

{

t1.seconds -= 60;

t1.minutes++;

}

while (t1.seconds < 0)

{

t1.seconds += 60;

t1.minutes--;

}

while (t1.minutes > 59)

{

t1.minutes -= 60;

t1.hours++;

}

while (t1.minutes < 0)

{

t1.minutes += 60;

t1.hours--;

}

while (t1.hours > 23)

{

t1.hours -= 24;

}

while (t1.hours < 0)

{

t1.hours += 24;

}

}

void error(const char field[])

{

cout << "Ошибка ввода " << field << ", попробуйте еще раз ";

char buf{};

cin.get(buf);

cin.ignore(32767, '\n');

}

Problem1.h:

#pragma once

#ifndef PROBLEM1\_H

#define PROBLEM1\_H

void problem1();

#endif // !PROBLEM1\_H

**Тест:**

***#1***

СЛЕД\_СЕК

Введите час: 23

Введите минуту: 59

Введите секунду: 0

Введите, на сколько секунд нужно перевести время: 4240

Переведенное время: 1:9:40

Введите 0, если хотите запустить программу еще раз:

***#2***

СЛЕД\_СЕК

Введите час: 0

Введите минуту: 0

Введите секунду: 0

Введите, на сколько секунд нужно перевести время: -4230

Переведенное время: 22:49:30

Введите 0, если хотите запустить программу еще раз:

***#3***

СЛЕД\_СЕК

Введите час: 24

Ошибка ввода часа, попробуйте еще раз

***#4***

СЛЕД\_СЕК

Введите час: 23

Введите минуту: -10

Ошибка ввода минуты, попробуйте еще раз

***#5***

СЛЕД\_СЕК

Введите час: 23

Введите минуту: 59

Введите секунду: 60

Ошибка ввода секунды, попробуйте еще раз

**Задача 1.2**

Ввести структуру для регистрации автомашин. Она должна иметь следующие поля: - дату регистрации (структура с полями - день, месяц, год);

- марку машины;

- год выпуска;

- цвет;

- номер.

Написать и протестировать функции:

- регистрация новой машины;

- удаление машины из регистрационного списка;

- поиск машины по любой из комбинаций признаков.

**Блок-схема:**









Листинг:

Problem2.cpp:

#include <iostream>

#include "Refresh.h"

using namespace std;

const int MAXLEN = 255;

const int BRAND = 50;

const int YEAR = 5;

const int COLOR = 15;

const int NUMBER = 10;

struct car

{

char\* date = new char[11] {"00.00.0000"};

char\* brand = new char[BRAND]{};

char\* year = new char[YEAR];

char\* color = new char[COLOR]{};

char\* number = new char[NUMBER]{};

};

void registration(car list[], int& len);

void output(car list[], int len);

void cardelete(car list[], int& len);

void carsearch(car list[], int len);

void problem2()

{

cout << "\tАвтомашины" << endl << endl;

car\* list = new car[MAXLEN];

int len = 0;

int com{};

while (com != 4)

{

cout << "Список команд:" << endl;

cout << " 0. Вывод списка" << endl;

cout << " 1. Добавление новой машины" << endl;

cout << " 2. Удаление машины" << endl;

cout << " 3. Поиск машины" << endl;

cout << " 4. Выход" << endl << endl;

cout << "Введите номер команды: ";

cin >> com;

refresh("Автомашины");

switch (com)

{

case 0:

output(list, len);

break;

case 1:

registration(list, len);

refresh("Автомашины");

break;

case 2:

cardelete(list, len);

cout << endl;

break;

case 3:

carsearch(list, len);

break;

}

}

delete[] list;

}

void registration(car list[], int &len)

{

car newcar;

cout << "Введите дату регистрации машины в формате ДД:ММ:ГГГГ: ";

cin >> newcar.date;

cout << "Введите марку машины: ";

cin >> newcar.brand;

cout << "Введите год выпуска машины: ";

cin >> newcar.year;

cout << "Введите цвет машины: ";

cin >> newcar.color;

cout << "Введите номер машины: ";

cin >> newcar.number;

cout << endl;

list[len] = newcar;

len++;

}

void output(car list[], int len)

{

cout << "Список машин: дата регистрации / марка / выпуск / цвет / номер" << endl;

for (int i = 0; i < len; i++)

{

cout << i << ". " << list[i].date << " / " << list[i].brand << " / " << list[i].year << " / " << list[i].color << " / " << list[i].number << endl;

}

cout << endl;

}

void cardelete(car list[], int& len)

{

output(list, len);

cout << "Введите номер машины, которую хотите удалить: ";

int num{};

cin >> num;

if (num >= len || num < 0)

{

cout << "Данный номер отсутствует" << endl;

}

else

{

for (int i = num; i < len - 1; i++)

{

list[i] = list[i + 1];

}

list[len] = {};

len--;

refresh("Автомашины");

}

}

void carsearch(car list[], int len)

{

car search{};

cout << "Заполните поля (если поле не используется в поиске, введите 0)" << endl;

cout << "Дата регистрации: ";

cin >> search.date;

cout << "Марка: ";

cin >> search.brand;

cout << "Год выпуска: ";

cin >> search.year;

cout << "Цвет: ";

cin >> search.color;

cout << "Номер: ";

cin >> search.number;

cout << endl << "Удалось найти: " << endl;

for (int i = 0; i < len; i++)

{

bool found = true;

if (strcmp(search.date, "0") && strcmp(list[i].date, search.date))

{

found = false;

}

if (strcmp(search.brand, "0") && strcmp(list[i].brand, search.brand))

{

found = false;

}

if (strcmp(search.year, "0") && strcmp(list[i].year, search.year))

{

found = false;

}

if (strcmp(search.color, "0") && strcmp(list[i].color, search.color))

{

found = false;

}

if (strcmp(search.number, "0") && strcmp(list[i].number, search.number))

{

found = false;

}

if (found)

{

cout << list[i].date << " / " << list[i].brand << " / " << list[i].year << " / " << list[i].color << " / " << list[i].number << endl;

}

}

cout << endl;

}

Problem2.h:

#pragma once

#ifndef PROBLEM2\_H

#define PROBLEM2\_H

void problem2();

#endif // !PROBLEM2\_H

**Тест:**

Автомашины

Список команд:

0. Вывод списка

1. Добавление новой машины

2. Удаление машины

3. Поиск машины

4. Выход

Введите номер команды:

***При вводе 1 и последующем заполнении:***

***Первый раз:***

Автомашины

Введите дату регистрации машины в формате ДД:ММ:ГГГГ: 28.02.2020

Введите марку машины: VAZ

Введите год выпуска машины: 2004

Введите цвет машины: Blue

Введите номер машины: B217RO

***Второй раз:***

Автомашины

Введите дату регистрации машины в формате ДД:ММ:ГГГГ: 15.05.2014

Введите марку машины: RENO

Введите год выпуска машины: 2014

Введите цвет машины: Blue

Введите номер машины: O108ER

***При вводе 0:***

Автомашины

Список машин: дата регистрации / марка / выпуск / цвет / номер

0. 28.02.2020 / VAZ / 2004 / Blue / B217RO

1. 15.05.2014 / RENO / 2014 / Blue / O108ER

Список команд:

0. Вывод списка

1. Добавление новой машины

2. Удаление машины

3. Поиск машины

4. Выход

Введите номер команды:

***При вводе 3 и последующем заполнении:***

Автомашины

Заполните поля (если поле не используется в поиске, введите 0)

Дата регистрации: 0

Марка: RENO

Год выпуска: 0

Цвет: 0

Номер: 0

Удалось найти:

15.05.2014 / RENO / 2014 / Blue / O108ER

Список команд:

0. Вывод списка

1. Добавление новой машины

2. Удаление машины

3. Поиск машины

4. Выход

Введите номер команды:

***При вводе 2 и последующем заполнении:***

Автомашины

Список машин: дата регистрации / марка / выпуск / цвет / номер

0. 28.02.2020 / VAZ / 2004 / Blue / B217RO

1. 15.05.2014 / RENO / 2014 / Blue / O108ER

Введите номер машины, которую хотите удалить: 1

***При вводе 0:***

Автомашины

Список машин: дата регистрации / марка / выпуск / цвет / номер

0. 28.02.2020 / VAZ / 2004 / Blue / B217RO

Список команд:

0. Вывод списка

1. Добавление новой машины

2. Удаление машины

3. Поиск машины

4. Выход

Введите номер команды:

***При вводе 2 и последующем заполнении:***

Автомашины

Список машин: дата регистрации / марка / выпуск / цвет / номер

0. 28.02.2020 / VAZ / 2004 / Blue / B217RO

Введите номер машины, которую хотите удалить: 3

Данный номер отсутствует

Список команд:

0. Вывод списка

1. Добавление новой машины

2. Удаление машины

3. Поиск машины

4. Выход

Введите номер команды:

**Задача 1.3**

Ввести структуру для описания комплексного числа. Составить и протестировать функцию для:

Преобразования комплексного числа из алгебраической формы в показательную.

**Блок-схема:**





Листинг:

Problem3.cpp:

#include <iostream>

using namespace std;

struct complex

{

float real{};

float imaginary{};

void algebraic()

{

cout << "z = " << real << " + " << imaginary << "i" << endl;

}

void indicative()

{

float mod = sqrt(real \* real + imaginary \* imaginary);

mod = round(mod \* 100) / 100;

float arg = atan(imaginary / real);

arg = round(arg \* 100) / 100;

cout << "z = " << mod << "e^" << arg << "i" << endl;

}

};

void problem3()

{

char res = '0';

while (res == '0')

{

cout << "\tКомплексное число" << endl << endl;

complex num;

cout << "Введите действительную часть числа: ";

cin >> num.real;

cout << "Введите мнимую часть числа: ";

cin >> num.imaginary;

cout << "Алгебраическая форма: ";

num.algebraic();

cout << "Показательная форма: ";

num.indicative();

cout << endl << "Введите 0, если хотите запустить программу еще раз: ";

cin.ignore(32767, '\n');

cin.get(res);

if (res == '0')

{

cin.get(res);

if (res != '\n')

{

res = '1';

}

else

{

res = '0';

}

}

else if (res != '\n')

{

cin.ignore(32767, '\n');

}

system("cls");

}

}

Problem3.h:

#pragma once

#ifndef PROBLEM3\_H

#define PROBLEM3\_H

void problem3();

#endif // !PROBLEM3\_H

**Тест:**

Комплексное число

Введите действительную часть числа: 12.3

Введите мнимую часть числа: 52.123

Алгебраическая форма: z = 12.3 + 52.123i

Показательная форма: z = 53.55e^1.34i

Введите 0, если хотите запустить программу еще раз:

**Задача 2.1**

Создать файл, содержащий сведения о книгах в библиотеке. Структура записи: шифр книги, автор, названия, год издания, местоположения (номер стеллажа, полка).

Предусмотреть возможность корректировки файла по вводимому коду корректировки, например:

1 - удалить запись (по шифру ХХХ);

2 - добавить новую запись;

3 - изменить запись (по введенной фамилии автора и названию книги);

4 - получить информацию о книге с шифром ХХХ.

**Блок-схема:**

















Листинг:

Problem4.cpp:

#include <iostream>

#include <fstream>

#include "Refresh.h"

using namespace std;

const int MAXLENOFFIELD = 50;

const int MAXLENOFLINE = 255;

const int CIPHER = 4;

void add(const char path[]);

void write(bool end, const char path[]);

void erase();

void showlist();

void change();

void info();

void problem4()

{

cout << "\tБиблиотека" << endl << endl;

int com{};

while (com != 5)

{

cout << "Список команд:" << endl;

cout << " 0. Список всех книг" << endl;

cout << " 1. Добавить книгу" << endl;

cout << " 2. Удалить книгу" << endl;

cout << " 3. Изменить запись" << endl;

cout << " 4. Получить информацию о книге" << endl;

cout << " 5. Выход" << endl << endl;

cout << "Введите команду: ";

cin >> com;

refresh("Библиотека");

switch (com)

{

case 0:

showlist();

cout << endl;

break;

case 1:

add("books.txt");

refresh("Библиотека");

break;

case 2:

erase();

refresh("Библиотека");

break;

case 3:

change();

refresh("Библиотека");

break;

case 4:

info();

cout << endl;

break;

}

}

}

void add(const char path[])

{

cin.ignore(32767, '\n');

cout << "Введите шифр (напр., 001): ";

write(false, path);

cout << "Введите автора: ";

write(false, path);

cout << "Введите название: ";

write(false, path);

cout << "Введите год издания: ";

write(false, path);

cout << "Введите стеллаж:полку (напр., 12:34): ";

write(true, path);

}

void erase()

{

cout << "Введите шифр книги, которую нужно удалить: ";

char\* cipher = new char[CIPHER] {};

cin >> cipher;

char\* line = new char[MAXLENOFLINE] {};

fstream file;

file.open("books.txt");

fstream temp;

temp.open("temp.txt", ios::out);

bool found = false;

file.getline(line, MAXLENOFLINE, '/');

while (!file.eof())

{

if (strcmp(line, cipher))

{

file.seekg(-4, ios\_base::cur);

file.getline(line, MAXLENOFLINE, '\n');

temp << line;

temp << '\n';

}

else

{

found = true;

file.getline(line, MAXLENOFLINE, '\n');

}

file.getline(line, MAXLENOFLINE, '/');

}

file.close();

temp.close();

if (found)

{

file.open("books.txt", ios::out);

temp.open("temp.txt");

temp.getline(line, MAXLENOFLINE, '\n');

while (!temp.eof())

{

file << line;

file << '\n';

temp.getline(line, MAXLENOFLINE, '\n');

}

file.close();

temp.close();

}

else

{

cout << "Книга не найдена" << endl;

char buf{};

cin.get(buf);

cin.ignore(32767, '\n');

}

delete[] cipher;

delete[] line;

}

void write(bool end, const char path[])

{

fstream file;

char\* str = new char[MAXLENOFFIELD] {};

cin.getline(str, MAXLENOFFIELD);

file.open(path, ios::app);

file << str;

end ? file << "\n" : file << "/";

file.close();

delete[] str;

}

void showlist()

{

fstream file;

file.open("books.txt");

char\* line = new char[MAXLENOFLINE] {};

file.getline(line, MAXLENOFLINE, '\n');

while (!file.eof())

{

cout << line << endl;

file.getline(line, MAXLENOFLINE, '\n');

}

}

void change()

{

cin.ignore(32767, '\n');

char\* author = new char[MAXLENOFFIELD] {};

char\* name = new char[MAXLENOFFIELD] {};

cout << "Введите автора: ";

cin.getline(author, MAXLENOFFIELD);

cout << "Введите название книги: ";

cin.getline(name, MAXLENOFFIELD);

char\* field = new char[MAXLENOFLINE] {};

char\* line = new char[MAXLENOFLINE] {};

fstream bks;

bks.open("books.txt");

fstream temp;

temp.open("temp.txt", ios::out);

fstream tempread;

tempread.open("books.txt");

bool found = false;

tempread.getline(line, MAXLENOFLINE, '\n');

while (!tempread.eof())

{

bks.getline(field, MAXLENOFLINE, '/');

bks.getline(field, MAXLENOFLINE, '/');

if (!strcmp(field, author) && !found)

{

bks.getline(field, MAXLENOFLINE, '/');

if (!strcmp(field, name))

{

found = true;

cout << "Запись найдена, введите новые значения" << endl;

temp.close();

add("temp.txt");

temp.open("temp.txt", ios::app);

}

else

{

temp << line << '\n';

}

}

else

{

temp << line << '\n';

}

bks.getline(field, MAXLENOFLINE, '\n');

tempread.getline(line, MAXLENOFLINE, '\n');

}

delete[] author;

delete[] name;

delete[] field;

bks.close();

temp.close();

if (found)

{

bks.open("books.txt", ios::out);

temp.open("temp.txt");

temp.getline(line, MAXLENOFLINE, '\n');

while (!temp.eof())

{

bks << line;

bks << '\n';

temp.getline(line, MAXLENOFLINE, '\n');

}

}

else

{

cout << "Книга не найдена" << endl;

char buf{};

cin.get(buf);

if (buf != '\n')

{

cin.ignore(32767, '\n');

}

}

delete[] line;

}

void info()

{

char\* cipher = new char[CIPHER] {};

cout << "Введите шифр: ";

cin >> cipher;

char\* str = new char[MAXLENOFLINE] {};

fstream file;

file.open("books.txt");

file.getline(str, MAXLENOFLINE, '/');

bool found = false;

while (!file.eof() && !found)

{

if (!strcmp(str, cipher))

{

found = true;

file.seekg(-4, ios\_base::cur);

file.getline(str, MAXLENOFLINE, '\n');

cout << str << endl;

}

file.getline(str, MAXLENOFLINE, '\n');

file.getline(str, MAXLENOFLINE, '/');

}

if (!found)

{

cout << "Книга не найдена" << endl;

}

}

Problem4.h:

#pragma once

#ifndef PROBLEM4\_H

#define PROBLEM4\_H

void problem4();

#endif // !PROBLEM4\_H

**Тест:**

Библиотека

Список команд:

0. Список всех книг

1. Добавить книгу

2. Удалить книгу

3. Изменить запись

4. Получить информацию о книге

5. Выход

Введите команду:

При вводе 0:

Библиотека

001/Pushkin/Anchar/1832/01:01

002/Dostoevsky/Idiot/1869/01:02

Список команд:

0. Список всех книг

1. Добавить книгу

2. Удалить книгу

3. Изменить запись

4. Получить информацию о книге

5. Выход

Введите команду:

При вводе 1 и последующем заполнении:

Библиотека

Введите шифр (напр., 001): 003

Введите автора: Gogol

Введите название: Dead souls

Введите год издания: 1842

Введите стеллаж:полку (напр., 12:34): 01:03

При вводе 2 и последующем заполнении:

Библиотека

Введите шифр книги, которую нужно удалить: 001

При вводе 3 и последующем заполнении:

Библиотека

Введите автора: Dostoevsky

Введите название книги: Idiot

Запись найдена, введите новые значения

Введите шифр (напр., 001): 002

Введите автора: Dostoevsky

Введите название: Crime and punishment

Введите год издания: 1866

Введите стеллаж:полку (напр., 12:34): 01:02

При вводе 0:

Библиотека

002/Dostoevsky/Crime and punishment/1866/01:02

003/Gogol/Dead souls/1842/01:03

Список команд:

0. Список всех книг

1. Добавить книгу

2. Удалить книгу

3. Изменить запись

4. Получить информацию о книге

5. Выход

Введите команду:

При вводе 4 и последующем заполнении:

Библиотека

Введите шифр: 002

002/Dostoevsky/Crime and punishment/1866/01:02

Список команд:

0. Список всех книг

1. Добавить книгу

2. Удалить книгу

3. Изменить запись

4. Получить информацию о книге

5. Выход

Введите команду:

При вводе 2 и последующем заполнении:

Библиотека

Введите шифр книги, которую нужно удалить: 001

Книга не найдена

При вводе 3 и последующем заполнении:

Библиотека

Введите автора: Pushkin

Введите название книги: Anchar

Книга не найдена

При вводе 4 и последующем заполнении:

Библиотека

Введите шифр: 001

Книга не найдена

Список команд:

0. Список всех книг

1. Добавить книгу

2. Удалить книгу

3. Изменить запись

4. Получить информацию о книге

5. Выход

Введите команду:

**Задача 2.2**

В файле находятся только целые числа. Определить, имеет ли последовательность чисел, находящихся в файле, нечетную длину, и если да то переменной middle присвоить значение среднего элемента файла. В противном случае присвоить этой переменной значение первого числа файла.

**Блок-схема:**



Листинг:

Problem5.cpp:

#include <iostream>

#include <fstream>

using namespace std;

const int MAXLEN = 255;

void problem5()

{

cout << "\tЧисла в файле" << endl << endl;

char\* middle = new char[MAXLEN] {};

int n = 0;

fstream file;

file.open("numbers.txt");

while (!file.eof())

{

file >> middle;

n++;

}

file.close();

file.open("numbers.txt");

if (n % 2 == 0)

{

file >> middle;

cout << "Кол-во чисел четное, первое число: " << middle << endl;

}

else

{

n /= 2;

for (int i = 0; i < n; i++)

{

file >> middle;

}

file >> middle;

cout << "Кол-во чисел нечетное, среднее число: " << middle << endl;

}

char buf{};

cin.get(buf);

if (buf != '\n')

{

cin.ignore(32767, '\n');

}

}

Problem5.h:

#pragma once

#ifndef PROBLEM5\_H

#define PROBLEM5\_H

void problem5();

#endif // !PROBLEM5\_H

**Тест:**

Содержимое numbers.txt: 1 22 333 55555 4444 123 000

Числа в файле

Кол-во чисел нечетное, среднее число: 55555

Содержимое numbers.txt: 1111 222 33 4

Числа в файле

Кол-во чисел четное, первое число: 1111

**Основная программа**

**Блок-схема:**





Листинг:

#include <iostream>

#include "Menu.h"

#include "Problem1.h"

#include "Problem2.h"

#include "Problem3.h"

#include "Problem4.h"

#include "Problem5.h"

using namespace std;

void main()

{

setlocale(LC\_ALL, "Russian");

char\* name = new char[54]{ "Обработка структур и файлов последовательного доступа" };

int nop = 5;

char\* p1 = new char[9]{ "СЛЕД\_СЕК" };

char\* p2 = new char[11]{ "Автомашины" };

char\* p3 = new char[18]{ "Комплексное число" };

char\* p4 = new char[11]{ "Библиотека" };

char\* p5 = new char[14]{ "Числа в файле" };

char\*\* ptrs = new char\* [sizeof(p1) + sizeof(p2) + sizeof(p3) + sizeof(p4) + sizeof(p5)]{ p1, p2, p3, p4, p5 };

void (\*fcnptrs[])() { problem1, problem2, problem3, problem4, problem5 };

menu(name, nop, ptrs, fcnptrs);

delete[] p1;

delete[] p2;

delete[] p3;

delete[] p4;

delete[] p5;

delete[] ptrs;

}

**Меню**

**Блок-схема:**





Листинг:

Menu.cpp:

#include <iostream>

using namespace std;

const int MAXLEN = 255;

void display(char progname[], int probs, char\* probname[]);

int digcheck(char str[]);

void menu(char progname[], int probs, char\* probname[], void (\*prog[])())

{

display(progname, probs, probname);

char\* input = new char[MAXLEN] {};

int num{};

while (num != probs)

{

cout << "Введите код: ";

do

{

cin >> input;

cin.ignore(32767, '\n');

num = digcheck(input);

if (num > probs || num < 0)

{

display(progname, probs, probname);

cout << "Неправильный код, попробуйте еще раз: ";

}

} while (num > probs || num < 0);

if (num != probs)

{

system("cls");

prog[num]();

display(progname, probs, probname);

}

}

delete[] input;

}

void display(char progname[],int probs, char\* probname[])

{

system("cls");

cout << "\t" << progname << endl << endl;

cout << "Задания:" << endl;

for (int i = 0; i < probs; i++)

{

cout << " " << i << ". " << probname[i] << endl;

}

cout << " " << probs << ". Выход" << endl;

}

int digcheck(char str[])

{

bool alldigit = true;

for (int i = 0; i < strlen(str); i++)

{

if (!isdigit(str[i]))

{

alldigit = false;

}

}

if (alldigit)

{

return atoi(str);

}

else

{

return -1;

}

}

Menu.h:

#pragma once

#ifndef MENU\_H

#define MENU\_H

void menu(char progname[], int probs, char\* probname[], void (\*prog[])());

#endif // !MENU\_H

**Дополнительные программы**

***Refresh***

**Блок-схема:**



Листинг:

Refresh.cpp:

#include <iostream>

using namespace std;

void refresh(const char title[])

{

system("cls");

cout << "\t" << title << endl << endl;

}

Refresh.h:

#pragma once

#ifndef REFRESH\_H

#define REFRESH\_H

void refresh(const char title[]);

#endif // !REFRESH\_H

**Тест:**

Обработка структур и файлов последовательного доступа

Задания:

0. СЛЕД\_СЕК

1. Автомашины

2. Комплексное число

3. Библиотека

4. Числа в файле

5. Выход

Введите код:

При вводе 0-4 происходит переход в программы задач, разобранных выше

При вводе 10, -4 или qwerty:

Обработка структур и файлов последовательного доступа

Задания:

0. СЛЕД\_СЕК

1. Автомашины

2. Комплексное число

3. Библиотека

4. Числа в файле

5. Выход

Неправильный код, попробуйте еще раз:

**Вывод:** были получены практические навыки работы с динамическими массивами, указателями, ссылками, структурами, файловым потоком, файлами последовательного доступа, создания .cpp и .h файлов, пользовательского меню, подключения собственных заголовочных файлов.