Министерство образования Республики Беларусь

Учреждение образования

«Брестский государственный технический университет»

Кафедра ИИТ

Лабораторная работа №5

По дисциплине: «ОАиП»

За 2 семестр

**Тема:** «Модульное программирование»

Выполнила:

Студентка 1 курса

Группы ПО-6(2)

Хурсина Е. П.

Проверил:

Хацкевич М. В.

Брест, 2021

Лабораторная работа №5

Модульно программирование

**Цель работы:** Изучить принципы модульного программирования в Си; ознакомиться с основными возможностями межмодульного взаимодействия.

**Задание:**

В программу, разработанную в лабораторной работе 12, внести следующие изменения и дополнения:

1. Программа должна быть разделена на несколько модулей .

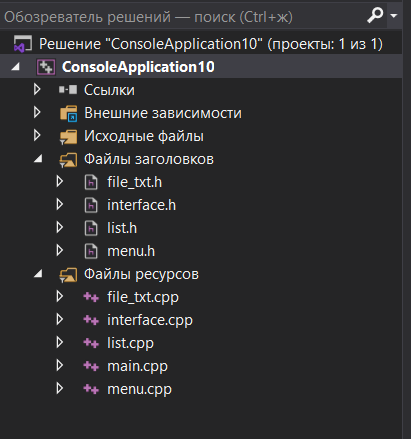
2. Взаимодействие модулей организовать при помощи вызова функций и переменных внешнего типа.

**Блок-схема**

Описание функции Void poisk(date\* date\_p, radio\* (&radio\_t), int& MAX). С помощью этой функции производиться поиск элемента массива до году сдачи в ремонт



**Текст программы:**



main.cpp – в данном модуле находится вызов функции меню.

file\_txt.h, file\_txt.cpp – модуль, который содержит функции, которые осуществляют действия с текстовыми файлами(ввод или запись в файл, чтение, добавление данных)

interface.h, interface.cpp – модуль, в который содержит функции, осуществляющие работу с массивом структур (сортировка, удаление, поиск)

list.h, list.cpp – модуль, который содержит функции, осуществляющие действия с массивом данных (ввод/вывод в консоль).

menu.h, menu.cpp – модуль, который содержит функцию для организации меню.

**main.cpp**

#include <windows.h>

#include "menu.h"

using namespace std;

int key;

int main() {

SetConsoleCP(1251);

SetConsoleOutputCP(1251);

system("color F0");

menu(file\_name);

return 0;

}

**file\_txt.h**

#pragma once

#include <fstream>

#include <iostream>

#include "list.h"

void write\_txt(texnika\* array, int MAX, ofstream& file); //запись данных

void read\_txt(texnika\*& array, int& MAX, char\* name); // чтение из файла

void rewrite(texnika\* index, int MAX); //перезапись даннных в файл

void index\_file(int MAX, int\* index); //индексные файлы

**file\_txt.cpp**

#include "file\_txt.h"

void rewrite(texnika\* array, int MAX, char\* name) {

ofstream file;

if (key == 1) {

file.open(name);

}

else if (key == 0) {

file.open(name, ios\_base::binary);

}

write\_txt(array, MAX, file);

file.close();

}

void write\_txt(texnika\* array, int MAX, char\* name) {

int k;

ofstream file\_txt;

if (key == 1) {

file\_txt.open(name);

}

else if (key == 0) {

file\_txt.open(name, ios\_base::binary);

}

ifstream index\_file;

index\_file.open("Index.txt");

if (key == 1) {

for (int i = 0; i < MAX; i++) {

int k;

if (index\_file.is\_open() && !index\_file.eof()) {

index\_file >> k;

index\_file >> k;

}

else k = i;

file\_txt << " ----------------------------------------------------------------------------------------------\n";

file\_txt << " | № | Наименование | Артикул | Дата сдачи | Состояние |\n";

file\_txt << " | | изделия | изделия | в ремонт | готовности |\n";

file\_txt << " ---------------------------------------------------------------------------------------------\n";

for (int t = 0; t < MAX; t++) {

k = t + 1;

if (array[t].radio\_t == 0) {

file\_txt << setw(3) << left << " | " << k << "| " << setw(3) << left << array[t].naim << " | " << setw(10) << left << array[t].num << array[t].date << " | " << setw(10) << left << array[t].radio\_t << setw(5) << left << " | " << endl;

file\_txt << " --------------------------------------------------------------------------------\n";

}

}

}

}

else if (key == 0) {

for (int i = 0; i < MAX; i++) {

int k;

if (index\_file.is\_open()) {

index\_file >> k;

index\_file >> k;

}

else k = i;

file\_txt.write((char\*)&array[k], sizeof(array[k]));

}

}

index\_file.close();

file\_txt.close();

}

**interface.h**

#pragma once

#include <iostream>

void sort(texnika\* array, int MAX, int key); //сортировка

int search(texnika\* array, int MAX); //поиск по заданному параметру

void deleted(texnika\*& array, int& MAX, char\* name); // удаление

**interface.cpp**

#include "interface.h"

void sort(texnika\* array, int MAX, int key) {

cout << "Выберите пункт:\n";

cout << "1 - Сортировка по наименовнию изделия\n";

cout << "2 - Сортировка по серийному номеру изделия\n";

int choise;

cin >> choise;

int\* index = new int[MAX];

for (int i = 0; i < MAX; i++) {

index[i] = i;

}

for (int i = 0; i < MAX - 1; i++) {

for (int j = i; j < MAX; j++) {

if (choise == 1) {

if (strcmp(array[index[j]].naim, array[index[i]].naim) < 0) {

swap(index[i], index[j]);

}

}

else if (choise == 0) {

if (array[index[j]].num < array[index[i]].num) {

swap(index[i], index[j]);

}

}

}

}

index\_file(MAX, index);

write\_txt(array, MAX, name);

list(array, MAX);

cin.ignore();

}

int search(texnika\* array, int MAX) {

int max = array[0].num, max\_ind = 0;

for (int i = 1; i < MAX; i++) {

if (array[i].num > max)

max = array[i].num;

}

return max;

}

void deleted(texnika\*& array, int& MAX, char\* name) {

char Name[20];

cout << "Введите наименование изделения, которое хотите удалит:.\n";

gets\_s(Name);

for (int i = 0; i < MAX; i++) {

if (strcmp(array[i].naim, Name) == 0) {

MAX--;

for (int j = i; j < MAX; j++)

array[j] = array[j + 1];

i--;

}

}

texnika\* tempArr = new texnika[MAX];

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

tempArr[i] = array[i];

array = new texnika[MAX];

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

array[i] = tempArr[i];

delete[] tempArr;

system("cls");

int\* tempArray = new int[MAX];

for (int i = 0; i < MAX; i++) {

tempArray[i] = i;

}

index\_file(MAX, tempArray);

delete[] tempArray;

write\_txt(array, MAX, name);

list(array, MAX);

}

**list.h**

#pragma once

#include <iostream>

#include <iomanip>

using namespace std;

enum texnica\_t { Бытовая\_техника = 1, Офисная\_техника, Медицинскя\_техника, Вычисдительная\_техника, Видеотехника, Красота\_здоровье };

union radio {

int num;

char access;

};

struct texnika {

char naim[20],

date[20];

int num;

radio radio;

unsigned radio\_t : 2;

};

void input(texnika\*& array, int& MAX); //ввод данных массива структуры

void list(texnika\* array, int MAX); //вывод данных массива структур

**list.cpp**

#include"list.h"

void input(texnika\*& array, int& MAX) {

int n;

cout << "Введите кол-во техники, которую сдали в ремонт: \n";

cin >> n;

MAX = MAX + n;

texnika\* tempArr = new texnika[MAX];

for (int i = 0; i < MAX - n; i++) {

tempArr[i] = array[i];

}

array = new texnika[MAX];

for (int i = 0; i < MAX - n; i++) {

array[i] = tempArr[i];

}

delete tempArr;

system("cls");

cin.ignore();

for (int k = 0; k < MAX; k++) {

cout << "\nВыберите категорию техники: \n1 -Бытовая\_техника \n2 - Медицинская\_техника \n3 - Офисная\_техника \n4 - Вычислительная\_техника \n5 - Видеотехника " << endl

<< "Ваш выбор: ";

int m;

cin >> m;

cout << "\nНаименование изделия: ";

cin >> array[k].naim;

cout << "\nСерийный номер изделия: ";

cin >> array[k].num;

cout << "Дата сдачи в ремонт.\n ";

cin >> array[k].date;

cout << "Состояние готовности заказа(1-выполнено/0- не выполнено): ";

std::cin >> array[k].radio\_t;

fflush(stdin);

}

}

void list(texnika\* array, int MAX) {

int k;

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

cout << " | № | Наименование | Артикул | Дата сдачи | Состояние |\n";

cout << " | | изделия | изделия | в ремонт | готовности |\n";

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

for (int t = 0; t < MAX; t++) {

k = t + 1;

if (array[t].radio\_t == 0) {

cout << setw(3) << left << " | " << k << "| " << setw(3) << left << array[t].naim << " | " << setw(10) << left << array[t].num << array[t].date << " | " << setw(10) << left << array[t].radio\_t << setw(5) << left << " | " << endl;

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

}

}

}

**menu.h**

#pragma once

#include "interface.h"

#include <iostream>

void menu(char\* file\_name); //организация меню

**menu.cpp**

#include "menu.h"

void menu(char\* file\_name) {

int choise, MAX = 0;

texnika\* array = new texnika[MAX];

cout << "Работать с текстовым(1) или бинарным файлом(0)?:\n";

cin >> key;

system("cls");

do {

cout << "Что вы хотите сделать?:\n";

cout << "1) Ввод данных\n";

cout << "2) Вывод данных\n";

cout << "3) Сортировка по алфавиту\n";

cout << "4) Поиск по серийному номеру\n";

cout << "5) Удалить запись\n";

cout << "6) Добавить новую запись\n";

cout << "7) Выход\n";

cin >> choise;

system("cls");

switch (choise)

{

case 1:

if (key == 1) {

read\_txt(array, MAX, name);

}

else if (key == 0) {

read\_bin(array, MAX);

}

break;

case 2:

cin.ignore();

list(array, MAX);

break;

case 3:

cin.ignore();

sort(array, MAX, key);

break;

case 4:

cin.ignore();

for (int i = 0; i < MAX; i++) {

if (array[i].num == search(array, MAX)) {

cout << "Искомая информация:\n";

cout << array[i].naim << " (" << array[i].num << endl;

}

}

break;

case 5:

cin.ignore();

deleted(array, MAX, name);

break;

case 6:

cin.ignore();

if (key == 1) {

add\_txt(array, MAX, name);

}

else if (key == 0) {

add\_bin(array, MAX);

}

break;

case 7: return 0;

}

cin.ignore();

system("cls");

} while (true);

delete[] array;

}

**Вывод:** в ходе данной лабораторной работы изучили принципы модульного программирования в Си; ознакомиться с основными возможностями межмодульного взаимодействия.

Модульное программирование – технология программирования, когда алгоритмы всей решаемой задачи разбиваются на отдельные, логически завершенные части.