МИНИСТЕРСТВО ОБРАЗОВАНИЯ РЕСПУБЛИКИ БЕЛАРУСЬ

Учреждение образования

«Гомельский государственный университет

имени Франциска Скорины»

Факультет математики и технологий программирования

Кафедра математических проблем управления и информатики

**Отчёт по контрольной работе**

**по дисциплине «Структуры и алгоритмы обработки данных»**

|  |  |
| --- | --- |
| Исполнитель  студент группы ПОз-41 \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ Э.О.Мохорев |  |
| Проверил:  Доцент кафедры МПУИ \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ А.Б.Демуськов |  |

Гомель 2023

Для прикладной области «Соревнование факультетов по успеваемости» разработать описание объектов этой области. Разработать процедуры, реализующие базовые операции над этими объектами:

* добавить запись,
* удалить запись,
* изменить запись (предусмотреть возможность редактирования всей записи и полей по выбору);

в том числе:

* текстовый ввод-вывод (консольный и файловый);
* присваивание;
* задание константных значений;
* сравнение (2 алгоритма сортировки – один для числового и второй для символьного поля).

Процедуры и описания данных должны составлять отдельный модуль (модуль типа данных).   
Используя процедуры и описания модуля типа данных, разработать программу, обеспечивающую ввод исходных данных из первого файла данных в память и хранение их в памяти в виде связного списка, сортировку списка по алфавитному и по числовому параметру.

**Код программы**

**kr.cpp**

#include <Windows.h>

#include "Operation.h"

int main()

{

SetConsoleCP(1251); //русский язык в консоли

SetConsoleOutputCP(1251); //русский язык в консоли

menu();

}

**Dannie.cpp**

struct Sorevnovanija {

char facultet[50];

int countStud;

double srBall;

int otl;

int dv;

};

struct Node {

Sorevnovanija sorevnovanija;

Node\* Next;

};

typedef

Node\* pNode;

**File Operation.cpp**

#define \_CRT\_SECURE\_NO\_WARNINGS

#include <Windows.h>

#include <stdio.h>

#include "Dannie.cpp"

typedef

Node\* pNode;

char file[200];

void newFile()

{

FILE\* F; //объявление переменной на файл

printf("Введите имя файла:"); //вывод надписи на экран

scanf("%s", file); //считать имя файла

fflush(stdin); //очистить буфер

strcat(file, ".zap"); //добавить расширение к имени

F = fopen(file, "wb"); //открыть файл для записи

fclose(F); //закрыть файл

F = NULL;

}

pNode openFile()

{

pNode Head = NULL; //указатель на начало списка NULL

FILE\* F; //указатель на файл

printf("Введите имя файла: "); //вод имени файла

scanf("%s", file); //считать имя

fflush(stdin); //очиситить буфер

strcat(file, ".zap"); //добавить к имени расширение

F = fopen(file, "rb"); //открыть файл для чтения записи

if (F == NULL) printf("Файл %s не найден или пуст\n", file); //если файл не открыт выдать ошибку

else

{

Sorevnovanija sorevnovanija; //переменная типа записи

pNode NewNode = new Node; //сздать новый узел

while (fread(&sorevnovanija, sizeof(Sorevnovanija), 1, F) != 0) //пока не конец файла, считывать в переменную записи

{

if (Head == NULL) { //если списка еще нет

Head = NewNode; //текущее начало списка соданный элемент

NewNode->Next = NULL; //следующего элемента нет

strcpy(NewNode->sorevnovanija.facultet, sorevnovanija.facultet); //занести значение

NewNode->sorevnovanija.countStud = sorevnovanija.countStud; //занести значение количества студентов

NewNode->sorevnovanija.srBall = sorevnovanija.srBall; //занести среднйий балл

NewNode->sorevnovanija.otl = sorevnovanija.otl; //занести отличников

NewNode->sorevnovanija.dv = sorevnovanija.dv; //занести двоечников

}

else //иначе

{

pNode q = new Node; //создать узел

strcpy(q->sorevnovanija.facultet, sorevnovanija.facultet); //занести значение

q->sorevnovanija.countStud = sorevnovanija.countStud; //занести количество

q->sorevnovanija.srBall = sorevnovanija.srBall; //занести средний балл

q->sorevnovanija.otl = sorevnovanija.otl; //отличников

q->sorevnovanija.dv = sorevnovanija.dv; //двоечников

q->Next = NULL; //следующий элемент за ним NULL

NewNode->Next = q; //добавить указатель в конец

NewNode = NewNode->Next; //перейти на доавленный элемент

}

}

fclose(F); //закрыть файл

}

return Head; //вернуть указатель на начало списка

}

void saveFile(pNode& Head)

{

pNode Node; //указатель на список

FILE\* F; //указатель на файл

if (strcmp(file, "") == 0) //если файл еще не открыт

{

printf("Введите имя файла: ");

scanf("%s", file); //считать имя файла

strcat(file, ".zap"); //добавить к нему расширение

}

F = fopen(file, "wb"); //открыть для записи

if (F == NULL) printf("Файл %s не найден или пуст\n", file); else //ели файл не открылся, выдать ошибку, иначе

{

Node = Head; //поставить указатель на начало списка

Sorevnovanija sorevnovanija; //переменная типа записи

while (Node != NULL) { //пока не конец списка

sorevnovanija = Node->sorevnovanija; //скопровать данные в переменную

fwrite(&sorevnovanija, sizeof(sorevnovanija), 1, F); //сохранить запись в файл

Node = Node->Next; //перейти на следующую запись

}

}

fclose(F); //закрыть файл

}

pNode CreatNode() {

pNode NewNode = new Node; //создать узел

int q; //целочисленная переменная

Sorevnovanija temp; //временная переменная типа структуры

printf("Введите факультет: "); //ввести новые данные

getchar(); //обнулитьь сивол \n в памяти

gets\_s(temp.facultet); //считать факультет (можно с пробелами)

printf("Введите количество студентов: ");

scanf("%d", &temp.countStud);//считать количество %d - целое число

printf("Введите средний балл: ");

scanf("%lf", &temp.srBall); //%lf - число типа double

printf("Введите количество отличников:");

scanf("%d", &temp.otl); //считать отличников

printf("Введите количество двоечников:");

scanf("%d", &temp.dv); //считать двоечников

NewNode->sorevnovanija = temp; //добавить данные в список

NewNode->Next = NULL; //следующий элемент

return(NewNode); //вернуть указатель на созданный элемент

}

pNode Add(pNode& Head)

{

pNode p = NULL, q = NULL; //указатели на списки

pNode NewNode = new Node; //создать новый

int r, l, i; //промежуточные переменные

q = Head; //присвоить начало спика

NewNode = CreatNode(); //вызвать функцию создания элемента

NewNode->Next = Head; //поменять ссылки на голову

Head = NewNode; //присвоить голове текущий элемент

return(Head); //вернуть голову

}

void vivodTop()

{

printf("\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\n");

printf("| № | Наименование факультета | Количество студентов | Средний балл | Отличники | Двоечники |\n");

printf("| п/п | | | | | |\n");

} //печать шапки таблицы

void vivodBot()

{

printf("|\_\_\_\_\_|\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_|\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_|\_\_\_\_\_\_\_\_\_\_\_\_\_\_|\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_|\_\_\_\_\_\_\_\_\_\_\_\_\_|\n");

} //печать подвала таблицы

void vivodMid(int k, pNode Node)

{

//вывести таблицу , %число обозначает сколько позиций отведено

printf("|%5d|%41s|%22d|%14.2lf|%15d|%13d|\n", k, Node->sorevnovanija.facultet, Node->sorevnovanija.countStud, Node->sorevnovanija.srBall, Node->sorevnovanija.otl, Node->sorevnovanija.dv);

}

void vivod(pNode& Head) {

pNode Node; //указатель на список

vivodTop(); //вывести шапку

Node = Head; //присвоить начало

int k = 1;

while (Node != NULL) { //если список не пуст

vivodBot(); //вывести подвал

vivodMid(k, Node); //вывод середины

Node = Node->Next; //переход к следующему элементу

k++; //увеличивать номер по порядку

}

vivodBot(); //вывод подвала

}

void deleteNode(pNode& Head)

{

pNode q = Head, OlmestNode; //указатели на список

int l, i; //временные переменные

vivod(Head); //вызов функции вывода всех записей

printf("Введите номер удаляемого элемента:");

scanf("%d", &l); //ввод номера по порядку удаляемой записи

i = 1; //начинать считать с единицы как пользователь

while (i != l) { //пока не пройдем нужное количество записей

q = q->Next; //перейти к следующей

i++; //увеличить счетчик

}

OlmestNode = q; //запомнить адрес нужного элемента

q = Head; //присвоить начало списка

if (Head == OlmestNode) //Если удаляется первый элемент

Head = OlmestNode->Next; //установить головой следующий за ним

else {

while (q->Next != OlmestNode) //иначе найти нужный элемент

q = q->Next; //перейти к следующей

if (q == NULL) return; //если не найден, то выйти

q = q->Next = OlmestNode->Next; //иначе переобозначить ссылки

}

delete OlmestNode; //удалить элемент

}

void EditNode(pNode& Head)

{

int l, k, i;

pNode Node;

Node = Head;

system("cls");

vivod(Head);

if (Head != NULL) {

printf("Введите номер записи для изменения:");

scanf("%d", &l);

printf("Введите номер поля для изменения:\n1 - Наименование факультета\n2 - Количество студентов\n3 - Средний былл\n4 - Количество отличников\n5 - Количество двоечников\n");

scanf("%d", &k);

i = 0;

while (Node != NULL) { //пока не найдена нужная запись

i++;

if (i == l) break; //если нашли, то выйти из уикла

Node = Node->Next; //иначе перейти дальше

}

if (Node != NULL) {

if (k == 1) { //если выбрано первое поле

printf("Введите наименование факультета:");

getchar(); //считать лишние символы из буффера

gets\_s(Node->sorevnovanija.facultet); //считать новое наименование

}

if (k == 2) {//если выбрано второе поле

printf("Введите количество студентов:");

fflush(stdin); //очистить буфер

scanf("%d", &Node->sorevnovanija.countStud); //считать количество студентов

}

if (k == 3) { //если выбрано третье поле

printf("Введите средний балл:");

fflush(stdin); //очистить буфер

scanf("%lf", &Node->sorevnovanija.srBall);//считать средний балл студентов

}

if (k == 4) { //если выбрано четвертое поле

printf("Введите количество отличников:");

fflush(stdin);//очистить буфер

scanf("%d", &Node->sorevnovanija.otl); //считать количество отличников

}

if (k == 5) {//если выбрано пятое поле

printf("Введите количество отличников:");

fflush(stdin); //очистить буфер

scanf("%d", &Node->sorevnovanija.dv); //считать новый диаметр

}

}

}

}

void sortStringFacult(pNode& Head){

/\* Cортировка вставками по Названию факультета \*/

pNode Node; //указатель на список

Sorevnovanija temp; //временная переменная типа структуры

pNode temp2; //указатель на список

Node = Head; //устанавливаем указатель на начало списка

int i = 0; //переменная счетчик

system("cls"); //очищаем экран

if (Head != NULL) { //если список не пуст

printf("Сортировка вставками по наименованию факультета");

pNode q, out, p, pr; //указатели на список

out = NULL; // выходной список пуст

while (Node != NULL) // пока не пуст входной список

{

q = Node; Node = Node->Next; // исключить очередной

// поиск места включения

for (p = out, pr = NULL; p != NULL && strcmp(q->sorevnovanija.facultet, p->sorevnovanija.facultet) > 0; pr = p, p = p->Next); //если первый больше второго

if (pr == NULL) // включение перед первым

{

q->Next = out; //переобозначить ссылки

out = q; //переоббозначить ссылки

}

else // иначе после предыдущего

{

q->Next = p; //переобозначить ссылки

pr->Next = q;//переобозначить ссылки

}

}

Head = out; //присвоить голове начало нового списка

}

}

void sortNodeNumber(pNode& Head) {

/\*Сортировка обменом для числового значения, сортирует по среднему баллу студентов.\*/

boolean changed = true; //флаг проверка на перестановку.

pNode Node; //указатель на список

Sorevnovanija temp; //временная переменная типа структуры

pNode temp2; //указатель на список

Node = Head; //устанавливаем указатель на начало списка

int i = 0; //переменная счетчик

system("cls"); //очищаем экран

if (Head != NULL) { //если список не пуст

printf("Сортировка среднего балла (сортировка простым обменом)");

while (Node != NULL && changed) { //пока не конец списка, но если массив отсортируется раньше то прервет.

changed = false;

temp2 = Node->Next; //присвоить следующий элемент

while (temp2 != NULL ) //если следующий существует

{

if (Node->sorevnovanija.srBall > temp2->sorevnovanija.srBall) //сравнить их, если первый больше второго обмен

{

temp = Node->sorevnovanija;

Node->sorevnovanija = temp2->sorevnovanija;

temp2->sorevnovanija = temp;

changed = true; // флаг который сигнализирует что произашел обмен

}

temp2 = temp2->Next; //перейти к следующему

}

Node = Node->Next; //перейти к следующему

}

}

}

void menu()

{

int q = 0; //переменная меню

pNode Head = NULL; //указатель на начало списка

pNode NewNode = NULL; //указатель на новый элемент

while (true) //бесконечный цикл

{

system("cls"); //очистить экран

printf(">>>>>>>>>>>>MENU<<<<<<<<<<<\n");

printf(" 1 - Новый файл\n");

printf(" 2 - Открыть файл\n");

printf(" 3 - Сохранить файл\n");

printf(" 4 - Добавить элемент\n");

printf(" 5 - Удаление записи\n");

printf(" 6 - Редактирование записи\n");

printf(" 7 - Сортировка по факультету (сортировка вставками)\n");

printf(" 8 - Сортировка по среднему баллу (сортировка обменом)\n");

printf(" 9 - Вывод базы данных\n");

printf("10 - Выход\n");

printf("Введите пункт меню:\n");

scanf("%d", &q); //считать пункт меню

switch (q) //в зависимости от пункта выполнить действия

{

case 1:

system("cls"); //очистить экран

newFile(); //вызвать функцию созадния нового файла

break;

case 2:

system("cls"); //очистить экран

Head = openFile(); //вызвать функцию открытия файла

break;

case 3:

system("cls"); //очистить экран

saveFile(Head); //вызвать функцию сохранения в файл

break;

case 4:

system("cls"); //очистить экран

Head = Add(Head); //вызвать функцию добавления

break;

case 5:

system("cls"); //очистить экран

deleteNode(Head); //вызвать функцию удаления

break;

case 6:

system("cls"); //очистить экран

EditNode(Head);//вызвать функцию редактирования

system("pause");//задержка экрана

break;

case 7:

system("cls"); //очистить экран

sortStringFacult(Head); //вызвать функцию сортировки

vivod(Head);//вызвать функцию вывода

system("pause");//задержка экрана

break;

case 8:

system("cls"); //очистить экран

sortNodeNumber(Head); //вызвать функцию сортировки

vivod(Head);//вызвать функцию вывода

system("pause");//задержка экрана

break;

case 9:

system("cls"); //очистить экран

vivod(Head);//вызвать функцию вывода

system("pause"); //задержка экрана

break;

case 10:

system("cls"); //очистить экран

exit(0); //выход

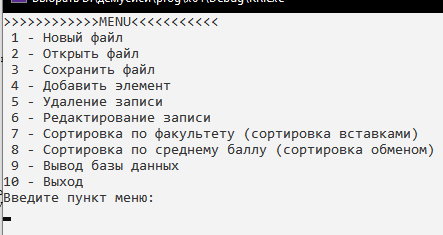
default: printf("Введено неверно"); //вывести ошибку

}

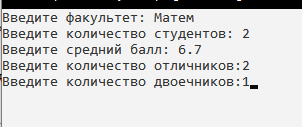
}

}

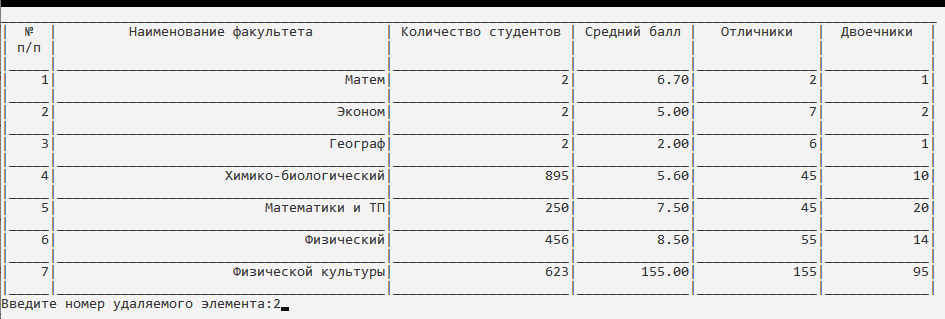
**Результат работы программы**



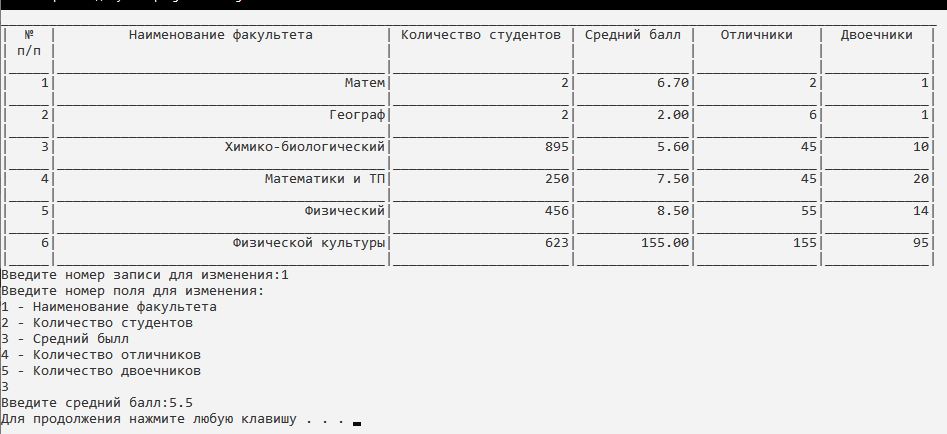
**Рисунок 1 – Меню**



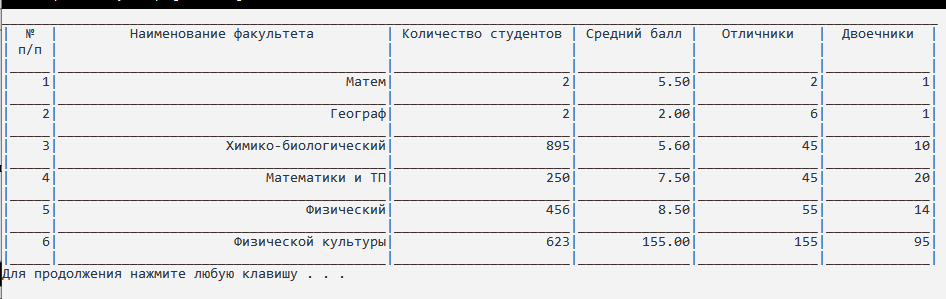
**Рисунок 2 – Добавление данных**



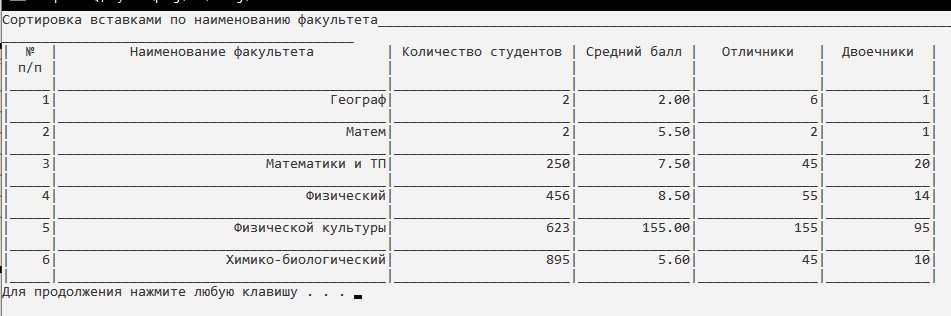
**Рисунок 3 - Удаление данных**



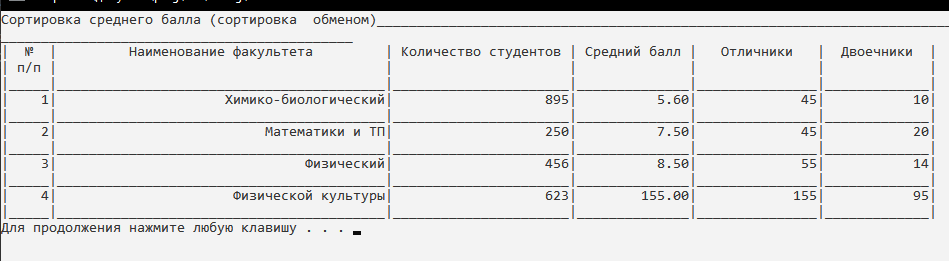
**Рисунок 4 - Изменение данных**



**Рисунок 5 - Вывод на экран**



**Рисунок 6 – Сортировка текстовых значений по факультету**



**Рисунок 7 – Сортировка числовых значений по среднему баллу**