**Лабораторная работа №2**

Схема БД. Меню программы

**Цель:** реализовать схему для работы с данными. Создать меню программы и функции для его реализации.

## Меню программы

Меню — элементом пользовательского интерфейса, позволяющим выбрать одну из нескольких перечисленных опций программы. Иногда же наличие меню является обязательным условием задания по программированию.  
Рассмотрим создание меню для консольной программы на примере следующей задачи: имеется список товаров, состоящих из наименования, цены и количества данного товара. Написать программу, позволяющую находить наименьшую и наибольшую цену. Взаимодействие с пользователем организовать с помощью меню.

Разбиваем программу на пункты  
Прежде чем добавлять меню в программу, следует разбить программу на процедуры, которые должны будут выполняться при выборе соответствующих пунктов меню. Нашу задачу можно разбить на следующие пункты:  
Добавление товара в список  
Вывод списка товаров (как минимум для контроля вводимых данных)  
Вывод товара с наибольшей ценой  
Вывод товара с наименьшей ценой  
Выход (иначе программа никогда не сможет завершиться)  
Выводим пункты меню на экран  
Чтобы вывести меню на экран, очистим его и затем просто напечатаем с помощью printf или cout номер пункта и то, что он должен делать:

void print\_menu() {  
system("cls"); // очищаем экран  
cout « "What do you want to do?" « endl;  
cout « "1. Add good to list" « endl;  
cout « "2. Print all goods in list" « endl;  
cout « "3. Print the highest price" « endl;  
cout « "4. Print the lowest price" « endl;  
cout « "5. Exit" « endl;  
cout « ">";  
}

Считываем введённый пункт меню  
В самом простом случае достаточно всего лишь считать число, однако правильнее будет защитить программу от некорректного ввода с помощью считывания строки и проверки ввода на корректность.

Вариант для C++:  
int get\_variant(int count) {  
int variant;  
string s; // строка для считывания введённых данных  
getline(cin, s); // считываем строку  
  
// пока ввод некорректен, сообщаем об этом и просим повторить его  
while (sscanf(s.c\_str(), "%d", &variant) != 1 || variant < 1 || variant > count) {  
cout « "Incorrect input. Try again: "; // выводим сообщение об ошибке  
getline(cin, s); // считываем строку повторно  
}  
  
return variant;  
}

Обработка выбранного пункта меню  
Создадим для каждого пункта свой обработчик в виде процедуры, выполняющей указанное действие:  
void add\_good(good \*\*goods, int \*size, int [\*capacity](https://vk.com/capacity)) {  
// реализация процедуры  
}  
  
void print\_goods(good [\*goods](https://vk.com/goods), int size) {  
// реализация процедуры  
}  
  
void print\_highest(good [\*goods](https://vk.com/goods), int size) {  
// реализация процедуры  
}  
  
void print\_lowest(good [\*goods](https://vk.com/goods), int size) {  
// реализация процедуры  
}  
Чтобы выполнить действие в зависимости от введённого пункта воспользуемся оператором switch-case:  
switch (variant) {  
case 1:  
add\_good(&goods, &size, &capacity);  
break;  
  
case 2:  
print\_goods(goods, size);  
break;  
  
case 3:  
print\_highest(goods, size);  
break;  
  
case 4:  
print\_lowest(goods, size);  
break;  
}  
Казалось бы, всё готово, однако пока что программа обработает лишь один пункт меню и затем завершится, хотя должна завершаться лишь при выборе пятого пункта - "выход". Чтобы организовать данный функционал, воспользуемся циклом do-while  
int variant;  
  
do {  
print\_menu();  
  
variant = get\_variant(5); // получаем номер выбранного пункта меню  
  
switch (variant) {  
case 1:  
add\_good(&goods, &size, &capacity);  
break;  
  
case 2:  
print\_goods(goods, size);  
break;  
  
case 3:  
print\_highest(goods, size);  
break;  
  
case 4:  
print\_lowest(goods, size);  
break;  
}  
  
if (variant != 5)  
system("pause"); // задерживаем выполнение, чтобы пользователь мог увидеть результат выполнения выбранного пункта  
} while (variant != 5);

Теперь наше меню будет работать до тех пор, пока пользователь не введёт число 5 чтобы выйти. При этом для всех остальных пунктов меню вызывается функция задержки system("pause"), чтобы после завершения работы выбранного пункта меню нашей консольной программы пункты меню выводились не сразу, а лишь тогда, когда пользователь самостоятельно решит вернуться в него.

## Вложенные структуры

Шаблон любой структуры может включать в себя другие структуры. Структура, являющаяся компонентом другой структуры, называется ***вложенной***. Экземпляр такой структуры был рассмотрен в примере 2 стр. 9-10.

***Для размещения вложенной структуры в шаблон*** она просто описывается точно так же, как это делалось бы для переменной типа *int*.

Например, есть шаблон структуры *Point*, который описывает точку на плоскости:

*struct Point*

*{*

*int x;*

*int y;*

*};*

Создадим шаблон *Triangle*, описывающий треугольник на плоскости:

*struct Triangle*

*{*

*Point p1; // вложенная структура*

*Point p2; // вложенная структура*

*Point p3; // вложенная структура*

*char comment[50]; // комментарий к фигуре*

*} tp;*

В шаблоне *Triangle* описывается три вложенных структуры (точки) *p1, p2, p3*, являющиеся переменнными типа *struct Point*.

**Получение доступа к элементу вложенной структуры**

Нужно дважды использовать операцию точка «•»:

tr**.**p1**.**x = 25;

tr**.**p2**.**x = 10;

tr**.**p3**.**x = 72;

Интерпретируем эту конструкцию, перемещаясь слева направо: (*tr.p1*).*x* –первым находим элемент *tr*, далее элемент *p1* структуры *tr*, а затем его элемент *x*.

#include <iostream>

using namespace std;

struct Point

{

int x;

int y;

};

struct Triangle

{

Point p1; // вложенная структура

Point p2; // вложенная структура

Point p3; // вложенная структура

char comment[50]; // комментарий к фигуре

} tr[] = {

{ 25, 45 },

{ 10, 86 },

{ 72, 48 },

"Координаты точек"

}**;**

void main(void)

{

setlocale(LC\_ALL, "Russian");

cout << tr.comment <<":" << endl;

cout << " x = " << tr.p1.x << '\t' << " y = " << tr.p1.y << endl;

cout << " x = " << tr.p2.x << '\t' << " y = " << tr.p2.y << endl;

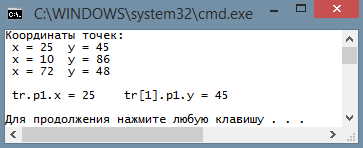
cout << " x = " << tr.p3.x << '\t' << " y = " << tr.p3.y << endl;

cout << endl;

// проверка

cout << " tr.p1.x = " << tr.p1.x << '\t' << " tr[1].p1.y = " << tr.p1.y << endl << endl;

}



# ПЕРЕДАЧА СТРУКТУР В ФУНКЦИЮ

**Структура** является **типом данных** созданным пользователем, к ней можно применять такие же операции, как и к встроенным типам. Рассмотрим работу структур (**struct**) с функциями.

1. Структуру можно передавать в функцию целиком:

#include <iostream>

#include <stdio.h>

using namespace std;

struct stud {

char name[20];

char group\_name[20];

float bal;

} **my[]** ={ "Липницкая И.А.", "УИР 2", 9.85,

"Скоморощенко К.С.", "УИР 1", 9.25,

"Щурок Е.А.", "УИР 3", 8.25,

}**;**

void print\_stud(**struct stud str**); //Прототип функции вывода

void main()

{

setlocale(LC\_ALL, "Russian");

for (int i = 0; i < 3; i++)

{

print\_stud(my[i]);

}

}

void print\_stud(struct stud str)

{

printf("ФИО : %15s\t", str.name);

printf("Группа : %s \t", str.group\_name);

printf(" Ср. балл : %.2f\n", str.bal);

}

1. Можно передать указатель на структуру.

void print\_stud(struct stud\* str); //Прототип функции вывода

void vvod\_stud(struct stud\* str, int n); //Прототип функции ввода

Код программы:

#include <iostream>

#include <stdio.h>

using namespace std;

struct stud {

char name[20];

char group\_name[20];

float bal;

};

void vvod\_stud(struct stud\* str, int n); // Размерность массива int n

void print\_stud(struct stud\* str);

void main()

{

setlocale(LC\_ALL, "Russian");

stud \*pmy = new stud [3] ;

vvod\_stud(pmy, 3);

/\*}\*/

print\_stud(pmy);

cout << endl;

}

void print\_stud(struct stud\* str)

{

for (int i = 0; i < 3; i++)

{

printf("ФИО : %15s\t", str->name);

printf("Группа : %s \t", str->group\_name);

printf(" Ср. балл : %.2f\n", str->bal);

str++;

}

}

void vvod\_stud(struct stud\* str, int n)

{

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

{

printf("\nВведите \n");

printf("ФИО : ");

gets(str[i].name);

printf("Группа : ");

gets(str[i].group\_name);

printf("Ср. балл : ");

cin >> str[i].bal;

cin.ignore();

}

}

1. Можно передать элементы структуры по отдельности:

void stud\_out(char name[20], float f)

{

printf("имя: %s; балл: %.1f\n", name, f);

}

void main()

{

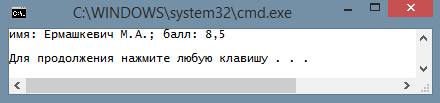
setlocale(LC\_ALL, "Russian");

struct stud a = { "Ермашкевич М.А.", "УИР 2", 8.5 };

stud\_out(a.name, a.bal);

cout << endl;

}



**Повторение:** При передаче самой структуры или ее элементов передаются значения. Т.е. при изменении элементов структуры в функции, настоящие элементы останутся нетронутыми. Для того, чтобы передавались настоящие элементы, в функцию надо передавать структуру или с помощью указателей, или с помощью ссылок.

***Пример:*** Создать структуру *Студент*: курс, группа, ФИО, какие предметы проходят, оценки за экзамен. Объявить динамический массив структур и заполнить его. Вывести на экран ведомость успеваемости. Передать структуру в функцию (ввод данных в массив структур).

Когда пользователь вводит данные в ответ на операцию извлечения, то эти данные помещаются в буфер ***std::cin***. Буфер данных – это просто часть памяти, зарезервированная для временного хранения данных, когда они перемещаются из одного места в другое. В этом случае буфер используется для хранения пользовательского ввода, пока он находится в режиме ожидании выделения для него переменных. В некоторых случаях возникает необходимость пропустить часть символов строки от начала до достижения конца строки (EOL) или конца файла (EOF). Именно этому и отвечает функция ***ignore()***. Она принимает два параметра: число пропускаемых символов и символ разделения. Например, вызов функции ignore(80, '\n') приведет к пропуску 80 символов, если ранее не будет найден символ начала новой строки. Последний затем будет удален из буфера, после чего функция ignore() завершит свою работу.

#include <stdlib.h>

#include <iostream>

#include <Windows.h>

using namespace std;

struct Stud1 //Объявление шаблона структуры Stud1

{

int Kurs;

int Gruppa;

char FIO[20];

char Predmet[30];

int Ocenka;

};

void GetData(Stud1 \*M, int N); //Прототип функции

void main() //Главная функция

{

setlocale(LC\_ALL, "Russian"); //Работа с кириллицей

SetConsoleCP(1251);

SetConsoleOutputCP(1251);

system("CLS"); //Очистка экрана

int i = 0;

int N;

cout << "Введите N: ";

cin >> N;

Stud1 \*M1 = new Stud1[N]; //Объявление динамического массива

// из N структур типа Stud1

GetData(M1, N); //Ввод данных в массив структур

cout << "\n";

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

{

if (M1[i].Ocenka <= 3)

{

cout << "Студент " << M1[i].FIO << " не сдал экзамен по предмету " << M1[i].Predmet << " : " << M1[i].Ocenka << endl << endl;

}

}

delete[]M1;

system("PAUSE");

}

void GetData(Stud1 \*M, int N) //Функция заполняет данными структуру Stud1

{

cin.ignore(); // Удаляем лишние значения

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

{

setlocale(LC\_ALL, "Russian");

cout << "\n";

cout << "Курс: ";

cin >> M[i].Kurs;

cout << "Группа: ";

cin >> M[i].Gruppa;

cin.ignore();

cout << "ФИО: ";

cin.getline(M[i].FIO, 20);

//gets(M[i].FIO); //Второй вариант ввода данных

cout << "Предмет: ";

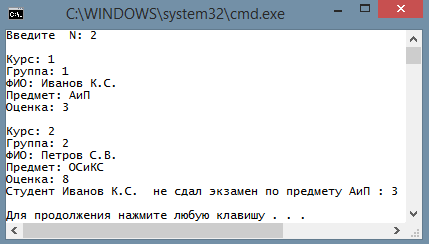
cin.getline(M[i].Predmet, 30);

cout << "Оценка: ";

cin >> M[i].Ocenka;

}

}



***Пример:*** Разработать программу, которая осуществляет работу с таблицей данных, содержащей информацию о студентах: №,Фамилия, Год рождения, Факультет. В программе использована структура. Проанализировать работу главной функции и функций, входящих в программу.

*# include <iostream>*

*# define str\_len 30*

*# define size 30*

*using namespace std;*

*int choice;*

*struct Student*

*{*

*char name[str\_len];*

*int year\_of\_birth;*

*char department[5];*

*};*

*struct Student list\_of\_student[size];*

*struct Student bad;*

*int current\_size = 0;*

*void enter\_new()*

*{*

*cout << "Ввод информации" << endl;*

*if (current\_size < size)*

*{*

*cout << "Строка номер ";*

*cout << current\_size + 1;*

*cout << endl << " \n Фамилия: " ;*

*cin >> list\_of\_student[current\_size].name;*

*cout << "\n Год рождения: " ;*

*cin >> list\_of\_student[current\_size].year\_of\_birth;*

*cout << "\n Факультет: " ;*

*cin >> list\_of\_student[current\_size].department;*

*current\_size++;*

*cout << endl;*

*}*

*else*

*cout << "Введено максимальное кол–во строк";*

*exit;*

*}*

*void del()*

*{*

*int d;*

*cout << "\nНомер строки, которую надо удалить (для удаления всех строк нажать 99)" << endl;*

*cin >> d;*

*if (d != 99)*

*{*

*for (int de1 = (d – 1); de1 < current\_size; de1++)*

*list\_of\_student[de1] = list\_of\_student[de1 + 1];*

*current\_size = current\_size – 1;*

*}*

*if (d == 99)*

*for (int i = 0; i < size; i++)*

*list\_of\_student[i] = bad;*

*exit;*

*}*

*void change()*

*{*

*int n, per;*

*cout << "\nВведите номер строки" << endl; cin >> n;*

*do*

*{*

*cout << "Введите: " << endl;*

*cout << "1–для изменения фамилии" << endl;*

*cout << "2–для изменения года рождения" << endl;*

*cout << "3–для изменения факультета" << endl;*

*cout << "4–конец\n";*

*cin >> per;*

*switch (per)*

*{*

*case 1: cout << "Новая фамилия";*

*cin >> list\_of\_student[n – 1].name; break;*

*case 2: cout << "Новый год рождения";*

*cin >> list\_of\_student[n – 1].year\_of\_birth; break;*

*case 3: cout << "Новый факультет ";*

*cin >> list\_of\_student[n – 1].department; break;*

*}*

*} while (per != 4);*

*exit;*

*}*

*void out()*

*{*

*int sw, n;*

*cout << "1–вывод 1 строки" << endl;*

*cout << "2–вывод всех строк" << endl;*

*cin >> sw;*

*if (sw == 1)*

*{*

*cout << "Номер выводимой строки " << endl; cin >> n; cout << endl;*

*cout << "Фамилия ";*

*cout << list\_of\_student[n – 1].name << endl;*

*cout << "Год рождения ";*

*cout << list\_of\_student[n – 1].year\_of\_birth << endl;*

*cout << "Факультет ";*

*cout << list\_of\_student[n – 1].department << endl;*

*}*

*if (sw == 2)*

*{*

*for (int i = 0; i < current\_size; i++)*

*{*

*cout << "Фамилия ";*

*cout << list\_of\_student[i].name << '\t';*

*cout << "Год рождения ";*

*cout << list\_of\_student[i].year\_of\_birth << '\t';*

*cout << "Факультет ";*

*cout << list\_of\_student[i].department << endl;*

*}*

*}*

*exit;*

*}*

*int main()*

*{*

*setlocale(LC\_CTYPE, "Russian");*

*cout << "Данных нет" << endl;*

*cout << "Введите:" << endl;*

*do*

*{*

*cout << "Введите:" << endl;*

*cout << "1–для удаления записи" << endl;*

*cout << "2–для ввода новой записи" << endl;*

*cout << "3–для изменения записи" << endl;*

*cout << "4–для вывода записи(ей)" << endl;*

*cout << "5–для выхода" << endl;*

*cin >> choice;*

*switch (choice)*

*{*

*case 1: del(); break;*

*case 2: enter\_new(); break;*

*case 3: change(); break;*

*case 4: out(); break;*

*}*

*} while (choice != 5);*

*}*

**ПРИМЕР:**

int main() {

setlocale(LC\_ALL, "Russian");

SetConsoleCP(1251);

SetConsoleOutputCP(1251);

int ch;

do {

setlocale(LC\_ALL, "Russian");

printf("\n1.Создать записи");

printf("\n2.Добавить запись");

printf("\n3.Вывести на экран");

printf("\n4.Количество записей");

printf("\n5.Найти запись");

printf("\n6.Изменить запись");

printf("\n7.Удалить запись");

printf("\n8.Сортировка в файле по баллу");

printf("\n9.Сортировка на консоли по баллу");

printf("\n10.Сортировка по имени");

printf("\n0.Выйти из программы");

printf("\nВведите свой выбор : ");

scanf("%d", &ch);

switch (ch) {

case 1:

create();

break;

case 2:

append();

break;

case 3:

display();

break;

case 4:

count();

break;

case 5:

search();

break;

case 6:

update();

break;

case 7:

delete\_rec();

break;

case 8:

sort\_in\_file();

break;

case 9:

sort\_on\_screen();

break;

case 10:

sort\_by\_name();

break;

case 0:

printf("\n\nСпасибо! До новых встреч!\n\n");

}

} while (ch != 0);

return 0;

}



#### ЗАДАНИЕ:

Определить шаблон структуры, содержащий информацию об однокурсниках и их лучших друзьях. Используйте вложенные структуры. Выполнить заполнение структуры данными через функцию. Организовать поиск информации и вывод результатов. Например, год рождения у лучшего друга однокурсника Иванова К.С., самый молодой однокурсник и его возраст.