Лабораторная работа №4.

Подпрограммы

Задание на лабораторную работу

Написать программу, реализующую хранение информации, указанной в вариантах индивидуальных заданий, в массиве структур и следующие действия:

- добавление элемента
- удаление элемента по заданному значению ключевого поля
- просмотр (вывод на экран) всех элементов, содержащих информацию
- поиск и изменение элемента по заданному значению ключевого поля
- просмотр (вывод на экран) всех элементов, у которых поле отбора имеет заданное значение
- Оформить все действия в виде функций, массив и другие данные передавать в функции как параметры.

Язык программирования С
Структуры

Структура – это структура данных, состоящая из фиксированного числа компонентов, называемых полями структуры. В отличие от массива, компоненты (поля) записи могут быт различного типа. Чтобы можно было ссылаться на тот или иной компонент записи, поля именуются. Структура является аналогом типа данных запись из языка Паскаль.

Переменные типа структура объявляются следующим образом:

```
struct <имя типа>
{
<список полей>
} <имя переменной>;
```

Здесь <имя типа>, <имя переменной> – правильный идентификатор;

struct – зарезервированное слово;

<список полей> – список полей; представляет собой последовательность разделов структуры, между которыми ставится точка с запятой.

Массив структур можно описать следующим образом:

```
struct Stud
{
   char fam[15], name [15];
   int group;
   Birthday bd;
   float rating;
} PMI[100];
```

Указатели

Язык Си имеет средства работы непосредственно с областями оперативной памяти ЭВМ, задаваемыми их адресами (указателями). В языке С указатели строго типизированы, т. е. различают указатели (адреса) символьных, целых, вещественных величин, а также типов данных, создаваемых программистом.

Для указателей одного и того же типа допустимой является операция присваивания, кроме того указателю типа void может быть присвоено значение адреса

данного любого типа, но не наоборот, например

```
int *a, *b;
double *d;
void *v;
...
a = b; /* Правильно */
v = a; /* Правильно */
v = d; /* Правильно */
b = v; /* Неправильно */
d = a; /* Неправильно */
```

Для поддержки адресной арифметики в языке Си имеются две специальные операции - операция взятия адреса & и операция получения значения по заданному адресу * (операция разадресации).

Рассмотрим работу вышеописанных операций на следующем примере

```
int *p, a, b;
double d;
void *pd;
p = &a;
*p = 12;
p = &b;
*p = 20;
/* Здесь a содержит число 12, b - число 20 */
pd = &d;
*( (double *) pd ) = a;
/* Здесь d содержит число 12.0 */
```

Состояние ячеек до первого присваивания

P, адрес 1000	a, адрес 2000	b, адрес 4000
мусор	мусор	мусор

Состояние ячеек после присваивания p = &a

p, адрес 1000	a, адрес 2000	b, адрес 4000
2000	мусор	мусор

Состояние ячеек после присваивания *p = 12

p, адрес 1000	a, адрес 2000	b, адрес 4000
2000	12	мусор

Состояние ячеек после присваивания p = &b

p, адрес 1000	a, адрес 2000	b, адрес 4000
4000	12	мусор

Состояние ячеек после присваивания *p = 20

p, адрес 1000	a, адрес 2000	b, адрес 4000
4000	12	20

Следует также опасаться случая, когда указатель содержит адрес объекта программы, завершившего свое существование. Например,

результат работы следующей программы неверен и непредсказуем:

```
#include <stdio.h>
#include <math.h>
double * Cube(double x)
{
  double cube_val;
  cube_val = x*x*x;
  return &cube_val;
}
void main(void)
{
  double *py;
  py = Cube(5);
  printf("y1 = %lf\n", *py);
  sin(0.7);
  printf("y1 = %lf\n", *py);
}
```