Лекция 7
Структуры в языке Си

Структуры в СИ

Структура – это сложный тип данных представляющий собой упорядоченное в памяти множество элементов различного типа. Каждый элемент в структуре имеет свое имя и называется полем. В СИ элементы структуры располагаются последовательно, а размер структуры определяется суммой размеров всех элементов.

Объявление в СИ структуры имеет вид:

```
struct  [имя типа]
{
   поле_1;
   поле_2;
   ...
   поле_N;
}  [список переменных];
```

Объявление полей структуры возможно только без инициализации. Если несколько полей следующих друг за другом в описании структуры имеют один и тот же тип, то для их описания можно использовать синтаксис объявления нескольких переменных одного и того же типа.

Структуры в языке С++ допускают вложенность, то есть использование структурной переменной в качестве поля какой-либо другой структуры.

Пример1 описания структуры

```
struct  part        // объявление структуры
{
   int modelnumber;     // номер модели изделия
   int partnumber;    // номер детали
   float cost;       // стоимость детали
};
part   part1, part2;   //  выделяем  память под
                        // переменные part1, part2
```

Доступ к полям структуры возможен с применением ***операции точки***. Например:

part1 . modelnumber = 6244;

Мы можем присваивать значение одной структурной переменной другой структурной переменной:

part2 = part1;

Важно! Эта операция присваивания может быть выполнена только над переменными, имеющими один и тот же тип. В случае попытки выполнить операцию присваивания над переменными разных типов компилятор выдаст сообщение об ошибке.

Пример2 описания структуры

Значения элементов структуры могут определяться **вводом**, **присваиванием**, **инициализацией**.

```
struct Student
{
   char suname[15];
   int kurs;
   char grup[6];
   float stip;
};
```

Пример инициализации в описании:

```
student  studl = {"Кротов", 1, “315a1", 350};

student  *pst,  stud1;
```

Тогда после выполнения оператора присваивания : **pst = &studl;**

доступ к полям структуры осуществляется следующим образом:

stud1.fam или **(*pst).fam** или **pst -> fam**

pst -> FIO, pst -> grup, pst -> stip.

Вложенные структуры

Структуры в языке C++ допускают вложенность, то есть использование структурной переменной в качестве поля какой-либо другой структуры.

```
struct Distance
{
   int feet;   //  футы
   float inches;   //   дюймы
};
struct Room
{
   Distance length;
   Distance width;
} dining;
```

Если одна структура вложена в другую, то для доступа к полям внутренней структуры необходимо дважды применить операцию точки (.):

```
dining . Length . feet  =  13;
```

В этом операторе **dining** — имя структурной переменной;

Length — имя поля внешней структуры **Room**;

feet — имя поля внутренней структуры **Distance.**

Таким образом, данный оператор берет поле **feet** поля **Length** переменной **dining** и присваивает этому полю значение, равное **13**.

Инициализация структурной переменной

Для инициализации структурной переменной, которая содержит внутри себя поле, также являющееся структурной переменной, значения, которыми инициализируется структурная переменная, заключаются в фигурные скобки и разделяются запятыми:

Room dining = { {13, 6.5} , {10, 0.0} };

Первая из внутренних структур инициализируется с помощью конструкции:

{13, 6.5}

а вторая – с помощью конструкции:

{10, 0.0}

Описание структуры в операторе typedef

От обязательного использования ключевого слова **struct** можно отказаться, если описывать структуру, используя оператор объявления типа **typedef** в следующем виде:

typedef struct [первичное имя типа] {...} имя типа;

Первичное имя типа, указываемое перед перечнем полей структуры является необязательным и указывается редко. Как правило, первичное имя типа имеет тот же идентификатор, что и основное имя типа, но начинается со знака подчеркивания.

Пример. Структура, содержащая информацию о книге (ФИО автора, название книги, год издания):

```
typedef struct {
   char author[20], title[50];
   unsigned year;
} BOOK;
```

Объявление переменной данного типа:

```
BOOK  book = {"Автор1”, ”Книга1”, 1986};
```

Размер переменной структурного типа.

Для определения размера переменной структурного типа в байтах используется оператор определения типа **sizeof**.

Например:

Структура, содержащая информацию о студенте (фамилия, имя, отчество, номер зачетной книжки, средний балл):

```
struct Student
{
    char sname[20], name[10], patronymic[20]; //  50  байт
    unsigned int num;  //  4 байт
    double mark;  //  8 байт
};

unsigned  size = sizeof (struct Student);   // size = 62
```

Примеры структур:

Структура, содержащая информацию о точке в двумерном пространстве (координаты):

```
struct Point
{
   double x, y;
};
```

Структура, содержащая информацию об окружности:

```
struct Circle
{
   double x, y, r;
};
```

Структура, содержащая информацию о студенте (фамилия, имя, отчество, номер зачетной книжки, средний балл):

```
struct Student
{
   char sname[20], name[10], patronymic[20];
   unsigned int num;
   double mark;
};
```

Структура, содержащая информацию о группе студентов (название группы, количество студентов, список студентов (максимально 30)):

```
struct Group
{
   char name[10];
   unsigned int number;
   struct Student list[30];
```

В языке СИ объявление переменной определенной структуры осуществляется после описания данной структуры в следующем виде:

struct тип имя_1 [= значение_1] [,...] ;

```
struct Point pnt[3] = {{0,0},{1,0},{0,1}};

struct Circle c1 = {10.0,10.0,5.0},
c2 = {0.0,0.0,25.0};

struct Student st = {"Иванов","Иван","
Иванович",959623,7.5};

struct Group gr =
 {
      "97-ВС",  3,
   {
{"Иванов", "Иван", "Иванович", 979601,
8.0},
{"Петров", "Петр", "Петрович", 979602,
6.5},
{"Сидоров","Сидор","
Сидорович",979603,9.0}
```

Обращение к полям структуры

Обращение к полям структуры осуществляется в следующем виде:

имя_переменной . имя_поля ;

Для обращения к конкретному полю структуры сначала пишется имя переменной структуры, а после через точку название поля. С точки зрения языка СИ при таком обращении к полю можно как получить его значение так и присвоить ему новое.

Примеры:

Вычисление длины окружности, заданной переменной cir типа Circle:

```
double length = 2.0*3.1415*cir.r;
```

Ввод информации о студенте в переменную st типа Student:

```
scanf ( "%s %s %s %u %lf", &st.sname, &st.name, &st.patronymic, &st.num, &st.mark);
```

Вывод на экран списка группы, заданной в переменой gr типа Group:

```
printf  ( "Группа: %s\n", gr.name);

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

  printf("%2u: %20s %20s %20s %6u %.1f\n",   i+1,  gr.list[i].sname,  gr.list[i].name,
                                              gr.list[i].patronymic, gr.list[i].num, gr.list[i].mark);
```

Массив структур. Пример.

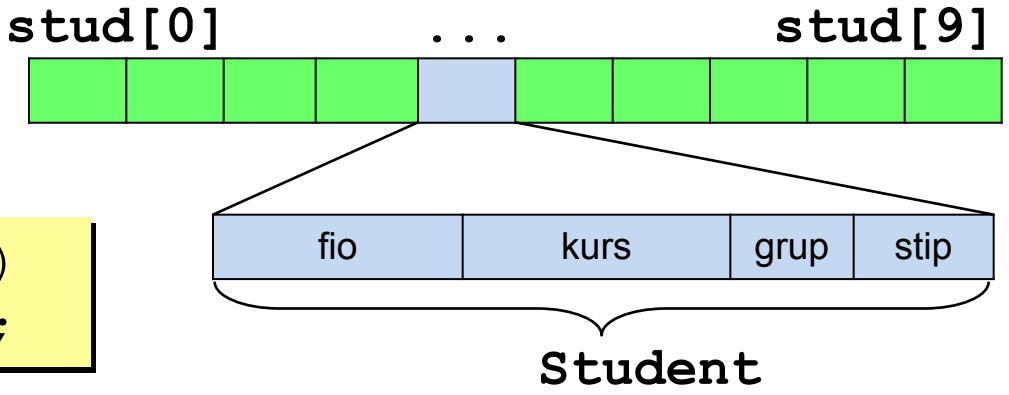
Задание. Ввести сведения об N студентах. Определить фамилии студентов, получающих повышенную стипендию.

```
#include <cstdio>
Struct  Student
{
char suname[15];
int kurs;
char grup[6];
float stip;
};
```

Массив структур

Объявление:

```
Student stud[10];
```



Обращение к полям:

```
for (i=0; i<10; i++)
  stud[i].grup=403a1;
```

Запись в *двоичный* файл:

```
FILE *f;
f=fopen("input.dat", "wb");
fwrite(stud, sizeof(Student), 10, f);
```

write binary

адрес массива

размер блока

сколько блоков

указатель на файл

Чтение из *двоичного* файла:

```
f=fopen("input.dat", "rb");
n=fread(stud, sizeof(Student), 10, f);
printf("Прочитано %d структур", n);
```

`fread` возвращает число удачно прочитанных блоков!

```
int main() {
    const int N = 5;
    int i,n;
    float maxs=0;
    Student stud[N];
    FILE *f;
    f = fopen("output.dat","wb");
    for(i=0; i<N; i++) {
        printf("%d-i student",i+1);
        printf("\nfio: ");
        scanf("%s",stud[i].suname);
        printf("\nkurs: ");
        scanf("%d",&stud[i].kurs);
        printf("\n№ gruppy: ");
        scanf("%s",stud[i].grup);
        printf("\nstipendiya: ");
        scanf("%f",&stud[i].stip);
    }
    fwrite(stud,sizeof(Student),N,f);
    fclose(f);
```

```
    for(i=0; i<N; i++) {
        if (stud[i].stip > maxs) {
            maxs = stud[i].stip;
        }
    }
    printf("\nstudenty s povyshennoi stipendiei %f rub.\n",maxs);
    for(i=0; i<N; i++) {
        if (stud[i].stip == maxs) {
            printf("%s\n",stud[i].fio);
        }
    }
    f = fopen("output.dat","rb");
    n = fread(stud,sizeof(Student),N,f);
    fclose(f);
    printf("\nprochitano %d struktur",n);
    return 0;
}
```

Выделение памяти под структуру

```
Student *p;
p=(Student*)malloc(sizeof(student));
printf("Фамилия ");
gets(p->suname);
printf(" Курс ");
scanf("%d",&p->kurs);
printf("Группа ");
gets(p->grup);
printf("Группа ");
scanf("%f",& p->stip);
...
free(p);
```

выделить память под структуру, записать ее адрес в переменную p

Для обращения к полю структуры по адресу используется стрелка ->!

освободить память

Задача

Допустим мы хотим хранить данные о компакт-дисках, которые имеются в нашей коллекции. Чем характеризуется диск? Прежде всего названием. Это конечно же строковый тип. Потом год выпуска. Следующий параметр - фирма изготовитель. Каждый диск записан какой-то фирмой, и упорядочив свои диски по фирме, мы сразу узнаем, какая фирма производит больше всего дисков, а какая фирма производит самые лучшие диски, то есть произведём некоторую статистику. Для аудиодисков немаловажно знать - сколько на нём песен. Ещё возможно мы хотим запомнить цену (тип float, так цена редко бывает не дробной) и краткую информацию - описание диска (например, какие на диске есть программы или какие представлены исполнители).

Задача. Сделать простейшую базу данных, которая будет хранить ваши данные в файле. Программа просит ввести данные для трёх дисков, а потом выводит их на экран.

При небольшой доработке: удобное меню, функция поиска, сохранение файла, увеличение количества записей, получится отличная база данных для хранения ваших данных.

Дана структура «Школьник», имеющая поля: фамилия, имя, пол, дата рождения (год, месяц, число), класс (цифра, буква). Напишите функцию чтения данных из файла в динамический массив структур. Напишите функцию поиска в динамическом массиве структур количества школьников, двенадцати лет, обучающихся в классах «Б».