华清创客学院
makeru.com.cn

C语言高级编程

主讲：小美老师

创客引领未来

扫微信二维码 获取更多信息

lemon

昵称：小美老师

华清创客学院，嵌入式讲师

maker

内容提纲

- 指针基础
- 指针和数组
- 多级指针
- const/void指针

内容提纲

- 函数定义、声明和调用
- 函数传参
- 指针函数和函数指针
- 递归函数

条件编译

- 编译器根据条件的真假决定是否编译相关的代码，
- 常见的条件编译有两种方法：

一、根据宏是否定义，其语法如下：

```
#ifdef <macro>
  ……
#else
  ……
#endif
```

条件编译

- 实例：

```
#define  _DEBUG_
#ifdef  _DEBUG_
printf(“The macro _DEBUG_ is defined\n”);
#else
printf(“The macro _DEBUG_ is not defined\n”);
#endif
```

条件编译

二、根据宏的值，其语法如下：

```
#if <macro>
 ……
#else
  ……
#endif
```

条件编译

- 实例：

```
#define  _DEBUG_   1
#if  _DEBUG_
printf(“The macro _DEBUG_ is defined\n”);
#else
printf(“The macro _DEBUG_ is not defined\n”);
#endif
```

课程目标

- 掌握C语言中结构体
- 掌握C语言中共用体

结构体

- 简述：
  - 在实际的处理对象中，有许多信息是由多个不同类型的数据组合在一起进行描述，而且这些不同类型的数据是互相联系组成了一个有机的整体。此时，就要用到一种新的构造类型数据——结构体（structure），简称结构。
  - 结构体的使用为处理复杂的数据结构（如动态数据结构等）提供了有效的手段，而且，它们为函数间传递不同类型的数据提供了方便。

结构体

- 概念
  - 结构体是用户自定义的新数据类型，在结构体中可以包含若干个不同数据类型和不同意义的数据项（当然也可以相同），从而使这些数据项组合起来反映某一个信息。
  - 例如，可以定义一个职工worker结构体，在这个结构体中包括职工编号、姓名、性别、年龄、工资、家庭住址、联系电话。这样就可以**用一个结构体数据类型的变量**来存放某个职工的所有相关信息。并且，用户自定义的数据类型worker也可以与int、double等基本数据类型一样，用来作为定义其他变量的数据类型

结构体

- 定义：
  - 定义一个结构体类型的一般形式为：

```
struct   结构体名
{
数据类型        成员名1;
数据类型        成员名2;
:
数据类型        成员名n;
};
```

结构体

- 在大括号中的内容也称为“成员列表”或“域表”。
- 其中，每个成员名的命名规则与变量名相同；
- 数据类型可以是基本变量类型和数组类型，或者是一个结构体类型；
- 用分号“；”作为结束符。整个结构的定义也用分号作为结束符

结构体

- Example：
  - 定义一个职工worker结构体如下：

```
struct worker
{
    long number;
   char name[20];
   char sex;
   int age;          //  age是成员名
   float salary;
   char address[80];
};                                          //注意分号不能省略
 int  age = 10;    //age是变量名
```

结构体

- 说明：

  结构体类型中的成员名可以与程序中的变量名相同，二者并不代表同一对象，编译程序可以自动对它们进行区分。

  最后，总结一下结构体类型的特点：

  （1）结构体类型是用户自行构造的。

  （2）它由若干不同的基本数据类型的数据构成。

  （3）它属于C语言的一种数据类型，与整型、实型相当。因此，定义它时不分配空间，只有用它定义变量时才分配空间。

结构体

- 结构体类型变量的定义方法

  先定义结构体类型再定义变量名

  这是C语言中定义结构体类型变量最常见的方式

```
struct 结构体名
{
        成员列表;
};
struct 结构体名 变量名;
```

结构体

- Example：

定义几个职工变量：struct worker

```
{
        long number;
        char name[20];
        char sex;
        int age;
        float salary;
        char address[80];
        char phone[20];
};
struct worker worker1,worker2;
```

结构体

- 注意事项：
  - “struct worker”代表类型名，不能分开写为：

```
struct worker1,worker2;
        //错误，没有指明是哪种结构体类型
或 worker worker1,worker2;
        //错误，没有struct关键字
        //系统不认为worker是结构体类型
```

  - 为了使用上的方便，程序员通常用一个符号常量代表一个结构体类型。即在程序开头加上下列语句：

```
#define WORKER struct worker;
```

这样在程序中，WORKER与struct worker完全等效。

结构体

- Example：

```
WORKER
{  long number;
   char name[20];
   char sex;
   int age;
   float salary;
   char address[80];
   char phone[20];  };
   WORKER worker1,worker2;
```

此时，可以直接用WORKER定义worker1、worker2两个变量，而不必再写关键字struct。

结构体

- 在定义类型的同时定义变量
  - 这种形式的定义的一般形式为：

```
struct 结构体名
{
    成员列表;
}变量名;
```

结构体

- Example:

```
struct worker
{   long number;
    char name[20];
    char sex;
    int age;
    float salary;
    char address[80];
    char phone[20];
} worker1,worker2;
```

结构体

- 直接定义结构类型变量
  - 其一般形式为：

```
struct        //没有结构体名
{
    成员列表;
}变量名;
```

结构体

- Example：

```
struct
{
    long number;
    char name[20];
    char sex;
    int age;
    float salary;
    char address[80];
    char phone[20];
} worker1,worker2;
```

结构体

- 大小

一个结构体变量占用内存的实际大小，也可以利用sizeof求出。它的运算表达式为：

sizeof（运算量）

//求出给定的运算量占用内存空间的字节数

其中运算量可以是变量、数组或结构体变量，可以是数据类型的名称。

例如：

```
sizeof(struct worker)
sizeof(worker1)
```

结构体

- 结构体变量的使用形式

结构体变量是不同数据类型的若干数据的集合体。在程序中使用结构体变量时，一般情况下不能把它作为一个整体参加数据处理，而参加各种运算和操作的是结构体变量的各个成员项数据。

结构体变量的成员用以下一般形式表示：

结构体变量名.成员名

例如，上节给出的结构体变量worker1具有下列七个成员：

worker1.number；worker1.name；worker1.sex；

worker1.age；worker1.salary；worker1.address；

worker1.phone

结构体

- 结构体变量的使用形式

在定义了结构体变量后，就可以用不同的赋值方法对结构体变量的每个成员赋值。例如：

```
strcpy(worker1.name,”Zhang San”);
worker1.age=26;
strcpy(worker1.phone,”1234567”);
worker1.sex=’m’;
        :
        :
```

除此之外，还可以引用结构体变量成员的地址以及成员中的元素。例如：引用结构体变量成员的首地址&worker1.name；引用结构体变量成员的第二个字符worker1.name[1]；引用结构体变量的首地址&worker1。

结构体

- 注意：

（1）不能将一个结构体类型变量作为一个整体加以引用，而只能对结构体类型变量中的各个成员分别引用。

例如，对上面定义的结构体类型变量wan，下列引用都是错误的：

```
cout<<wan;
cin>>wan;
```

但是可以如下引用：

```
cout<<wan.name;
cin>>wan.name;
```

结构体

（2）如果成员本身又属一个结构体类型，则要用若干个成员运算符，一级一级地找到最低的一级成员。只能对最低级的成员进行赋值或存取以及运算。例如，对上面定义的结构体类型变量worker1，可以这样访问各成员：

```
worker1.age
worker1.name
worker1.birthday.year
worker1.birthday.month
worker1.birthday.day
```

注意：不能用worker1.birthday来访问worker1变量中的成员birthday，因为birthday本身是一个结构体变量。

结构体

（3）对成员变量可以像普通变量一样进行各种运算（根据其类型决定可以进行的运算）。例如：

```
worker2.age=worker1.age;
sum=worker1.age+worker2.age;
worker1.age++;
```

（4）在数组中，数组是不能彼此赋值的，而结构体类型变量可以相互赋值。

在C程序中，同一结构体类型的结构体变量之间允许相互赋值，而不同结构体类型的结构体变量之间不允许相互赋值，即使两者包含有同样的成员。

结构体

- 结构体变量的初始化

与其他类型变量一样，也可以给结构体的每个成员赋初值，这称为结构体的初始化。一种是在定义结构体变量时进行初始化，语法格式如下：

```
struct  结构体名 变量名={初始数据表};
```

另一种是在定义结构体类型时进行结构体变量的初始化。

```
struct  结构体名
{
        成员列表;
}变量名={初始数据表};
```

结构体

- 前述student结构体类型的结构体变量wan在说明时可以初始化如下：

struct student wan={”Wan Jun”,’m’,20,” SuZhou Road No.100”};

- 等价于下列代码：

strcpy(wan.name,” Wan Jun”);
wan.sex = ’m’;
wan.age = 20;
strcpy(wan.addr, ” SuZhou Road No.100”);

扫微信二维码 获取更多信息