第九章 结构体

郎大鹏

第九章 结构体

9.1　结构体类型的声明方法

结构体声明的语法形式如下：

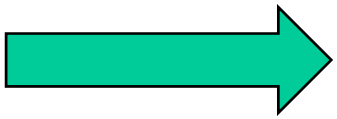
struct　结构体标识符

{

成员变量列表;

…

};



例如，为了描述班级（假设仅仅包括班级编号、专业、人数等信息），可以声明如下的结构体类型。

```
struct Class
{
char Code[10]; /*编号 */
char Major[30]; /*专业*/
unsigned int Count; /*人数*/
};
```

9.2 结构体类型变量的定义与使用

9.2.1 变量的三种定义方法

1．先声明结构体类型，再定义变量

先声明一个结构体类型，其中student为结构体类型名，如：

```
struct student                /*该类型共包含4个成员，各成员的类型可以不同*/
{
int num;
    char name[20];
    int age;
    float score;
};
                              /*作为结构体类型的声明，此行最后的“；”不可缺少*/
…
struct student stud1,stud2；   /*使用时定义两个属于struct student类型的结构体变量stud1和
stud2。其中struct student为结构体类型名*/
```

9.2 结构体类型变量的定义与使用

9.2.1 变量的三种定义方法

2．在声明结构体类型的同时定义变量

如：struct student

```
    {
int num;
        char name[20];
        int age;
        float score;
    }stud1,stud2;        /* 定义了两个struct student类型的变量stud1、stud2*/
```

这种情况直接将结构体变量stud1和stud2放在声明结构体类型时的}和;之间定义。

9.2 结构体类型变量的定义与使用

9.2.1 变量的三种定义方法

3．直接定义结构体类型变量

如：struct /* 结构体名并不出现，而直接定义变量*/

```
{
int  num；
        char name[20]；
        int age；
        float score；
      }stud1,stud2；
```

与第一种和第二种方法相比，若要再定义一些与stud1、stud2相同类型的结构体变量时，这种方法显然不方便。

9.2 结构体类型变量的定义与使用

关于结构体类型的几点说明:

- 类型与变量是不同的概念，不能混淆;
- 可以对变量进行存取、赋值等运算;
- 却不能对类型进行这些操作或运算;
- 在编译时只对变量分配空间，并不对类型分配空间;
- 可以单独使用结构体变量中的成员，其地位和作用相当于普通变量。

9.2.2　结构体变量的初始化

简单变量的初始化形式如下：

数据类型　变量名=初始化值；

例如，定义整型变量a，并给其初始化值10的语句如下：

```
int a=10;
```

数组的初始化，需要通过一常量数据列表，对其数组元素分别进行初始化，形式如下：

数据类型　数组名称［数组长度］={初始化值1，初始化值2，… ，初始化值n}；

例如，定义长度为5的整型数组，并对其初始化的语句如下：

```
int A[5]={20,21,0,3,4};
```

9.2.2　结构体变量的初始化

结构体变量的初始化方式与数组类似，分别给结构体的成员变量赋以初始值，而结构体成员变量的初始化遵循简单变量或数组的初始化方法。具体的形式如下：

```
struct    结构体标识符
{
    成员变量列表;
    …
};
struct  结构体标识符   变量名={初始化值1，初始化值2，… ，初始化值n };
```

9.2.2 结构体变量的初始化

表9.1 基本数据类型成员变量的初始化缺省值

数据类型	缺省初始化值
int	0
char	‘\0x0’
float	0.0
double	0.0
char Array[n]	""
int Array[n]	{0,0…,0}

9.2.2 结构体变量的初始化

例如：

```
struct Line
{
    int id;
    struct Point StartPoint;
    struct Point EndPoint;
}
oLine1={
    0, /*初始化id */
    {0,0,0}, /*初始化StartPoint*/
    {100,0,0} /*初始化EndPoint */
};
```

9.2.3　结构体变量的使用

1．结构体成员变量的引用

结构体变量包括一个或多个成员变量，引用其成员变量的语法格式如下：

结构体变量名.成员变量名

例如，有如下的定义：

struct Point oP1={0.0,0.2,0.3};

结构体struct Point具有三个成员，

double x;

double y;

double z;

9.2.3　结构体变量的使用

2．结构体变量本身的引用

结构体变量本身的引用是否遵循基本数据类型变量的引用规则呢？例如下面的程序：

```
struct Point oP1={0.0,0.2,0.3};
struct Point oP2,oP3;
oP2= oP1;
oP3= oP1+ oP2*2;
```

在C语言的语法中，这样引用结构体类型变量是违法的。在C语言定义的所有的运算符中，只有运算符. 和&以及sizeof可以用于结构体变量，其中

“.” ——用于获得成员变量，如oP1.x。

“&” ——用于获得变量地址，如&oP1。

“sizeof” ——用于获得变量存储空间的大小，单位为字节，如sizeof(oP1)。

9.2.3　结构体变量的使用

例9.1　输入10个同学的姓名、数学成绩、英语成绩和物理成绩，确定总分最高的同学，并打印其姓名及其三门课程的成绩。

```
#include "stdio.h"
struct Student /*定义结构体struct Student*/
{
    char Name[20]; /*姓名*/
    float Math; /*数学*/
    float English; /*英语*/
    float Physical; /*物理*/
};
```

```
void main()
{
struct Student oStu;
struct Student oMaxStu;
int i;
float fMaxScore;
float fTotal;
printf("\nPlease input 10 students and there
scores\n");/*提示信息*/
printf("---------------------------------\n");
printf("Name Math English Physical \n");
printf("---------------------------------\n");
fMaxScore=0;
```

```
for(i=0;i<10;i++){/*读入当前同学的相关信息*/
scanf("%s %f %f
%f",oStu.Name,&oStu.Math,&oStu.English,&oSt
u.Physical);
fTotal=oStu.Math+oStu.English+oStu.Physical;
if(fMaxScore<fTotal)
{ fMaxScore=fTotal; /*保存当前最大成绩和*/
/*保存当前最大成绩和的同学的相关信息*/
strcpy(oMaxStu.Name,oStu.Name);
oMaxStu.Math=oStu.Math;
oMaxStu.English=oStu.English;
oMaxStu.Physical=oStu.Physical;
}}
printf("---------------------------------\n");
printf("%s %f %f %f \n ",
oMaxStu.Name,oMaxStu.Math,oMaxStu.English
,oMaxStu.Physical);
}
```

9.2.4　结构体变量指针

1．结构体指针变量的定义

定义结构体指针变量的一般形式如下：

形式1：

struct　结构体标识符

{

成员变量列表;

…

};

struct 结构体标识符 *指针变量名;

形式2：

struct　结构体标识符

{

成员变量列表;

…

} *指针变量名;

形式3：

struct

{

成员变量列表;

…

}*指针变量名;

9.2.4　结构体变量指针

2．结构体指针变量的初始化

结构体指针变量在使用前必须进行初始化，其初始化的方式与基本数据类型指针变量的初始化相同，在定义的同时赋予其一结构体变量的地址。例如

```
struct Point oPoint={0,0,0};
struct Point *pPoints=& oPoint; /*定义的同时初始化*/
```

在实际应用过程中，可以不对其进行初始化，但是在使用前必须通过赋值表达式赋予其有效的地址值。例如：

```
struct Point oPoint={0,0,0};
struct Point *pPoints2;
pPoints2=& oPoint; /*通过赋值表达式*/
```

9.2.4　结构体变量指针

3．结构体指针变量的引用和运算

与基本类型指针变量相似，结构体指针变量主要作用是存储其结构体变量的地址或结构体数组的地址，通过间接方式操作对应的变量和数组。在C语言中规定，结构体指针变量可以参与的运算符如下：

++，--，+，*，->，.，|，&，！

下面通过例题说明，如何引用结构体指针变量存储结构体变量地址，以及如何通过结构体指针变量间接的引用结构体变量以及其成员变量。

9.2.4　结构体变量指针

例9.2　应用结构体指针变量，打印结构体成员变量的信息。

```
#include "stdio.h"
struct Point
{
double x; /*x坐标*/
double y; /*y坐标*/
double z; /*z坐标*/
};
```

```
int main()
{
struct Point oPoint1={100,100,0};
struct Point oPoint2;
struct Point *pPoint; /*定义结构体指针变量
*/
pPoint=& oPoint2;/*结构体指针变量赋值*/
(*pPoint).x= oPoint1.x;
(*pPoint).y= oPoint1.y;
(*pPoint).z= oPoint1.z;
printf("oPoint2={%7.2f,%7.2f,%7.2f}
\n",oPoint2.x,oPoint2.y,oPoint2.z);
return(0);
}
```

9.2.4　结构体变量指针

通过结构体指针变量获得其结构体变量的成员变量的一般形式如下：

(*结构体指针变量).成员变量

其中“结构体指针变量”为结构体指针变量名，“成员变量”为结构体成员变量名，“.”为取结构体成员变量的运算符。

另外C语言中引入了新的运算符“->”，通过结构体指针变量直接获得结构体变量的成员变量，一般形式如下：

结构体指针变量->成员变量

其中“结构体指针变量”为结构体指针变量名，“成员变量”为结构体成员变量名，“->”为运算符。

9.3 结构体数组

9.3.1 结构体数组的定义

数组是一组具有相同数据类型变量的有序集合，可以通过下标获得其中的任意元素。结构体类型数组与基本类型数组的定义与引用规则是相同的，区别在于结构体数组中的所有元素均为结构体类型变量。

9.3.1　结构体数组的定义

方式1：

```
struct　结构体标识符
{
成员变量列表;
…
};
…
struct　结构体标识符　数组名[数组长度];
```

方式2：

```
struct　结构体标识符
{
成员变量列表;
…
}数组名[数组长度];
```

方式3：

```
struct
{
成员变量列表;
…
}数组名[数组长度];
```

9.3.1 结构体数组的定义

例：定义长度为**10**的**struct Point**类型数组**oPoints**的方法有如下三种形式：

方式1：

```
struct Point
{
double x;
double y;
double z;
};
…
struct Point oPoints[10];
```

方式**2**：

```
struct Point
{
double x;
double y;
double z;
} oPoints[10];
```

方式**3**：

```
struct
{
double x;
double y;
double z;
} oPoints[10];
```

9.3.2 结构体数组的初始化

结构体类型数组的初始化遵循基本数据类型数组的初始化规律，在定义数组的同时，对其中的每一个元素进行初始化。例如：

```
struct Student /*声明结构体struct Student*/
{
char Name[20]; /*姓名*/
float Math; /*数学*/
float English; /*英语*/
float Physical; /*物理*/
}oStus[2]={
{"Liming",78,89,95},
{"Majun",87,79,92}
};
```

9.3.2 结构体数组的初始化

在定义数组并同时进行初始化的情况下，可以省略数组的长度，系统根据初始化数据的多少来确定数组的长度。例如：

```
struct Key
{
char word[20];
int count;
}keytab[]={
{"break",0},
{"case",0},
{"void",0}
};
```

结构体数组keytab的长度，系统自动确认为3。

9.3.3　结构体数组的使用

1．结构体数组元素的引用

结构体数组元素引用的语法形式如下：

结构体数组名[数组下标];

[]为下标运算符，数组下标的取值范围为（0，1，2，...，n-1），n为数组长度。

9.3.3　结构体数组的使用

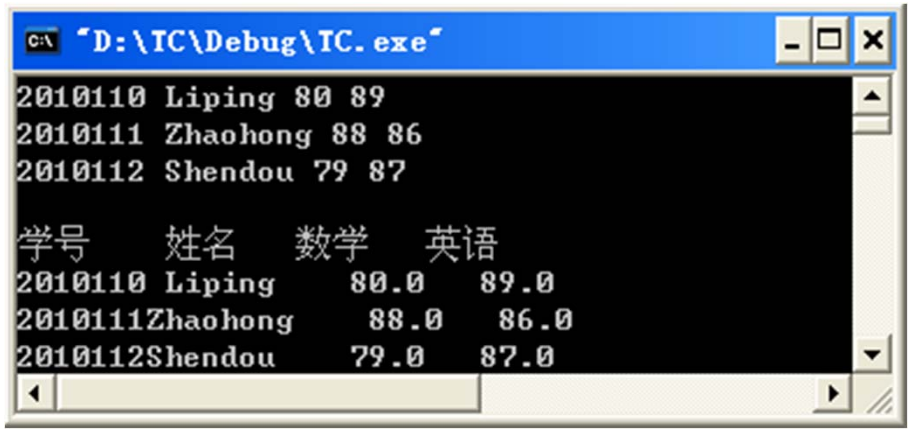
例9.3　某班有若干名学生，每一位学生的信息包括学号、姓名、2门课程的成绩。定义一个可以存放3个学生的结构体数组，实现对结构体数组的输入与输出。

```
#include "stdio.h"
struct student{
 // 声明结构体类型

long int number;        // 学号
char name[8];           // 姓名

 float score[2];        // 2门课
程的成绩
};
```

9.3.3 结构体数组的使用




```
void main()
{
struct student stud[3];      //定义结构体数组
      int i,j;
      //输入
      for(i=0;i<3;i++)
      {
            scanf("%ld",&stud[i].number);
            scanf("%s",stud[i].name);
            for(j=0;j<2;j++)
            scanf("%f",&stud[i].score[j]);}
     //输出
     printf("\n学号   姓名   数学   英语\n");
     for(i=0;i<3;i++)
     {
printf("%ld",stud[i].number);
printf("%7s ",stud[i].name);
for(j=0;j<2;j++)
printf("%7.1f\n",stud[i].score[j]);
}
```

9.3.3　结构体数组的使用

2．结构体数组的引用

结构体数组作为一个整体的引用，一般表现在如下两个方面：

（1）作为一块连续存储单元的起始地址与结构体指针变量配合使用，此问题可参考结构体指针部分。

（2）作为函数参数。函数的形式参数为结构体类型数组，在调用函数时将实际参数即定义好的结构体数组名作为整体传入

9.3.3　结构体数组的使用

例9.4　某班有若干名学生，每一位学生的信息包括学号、姓名、2门课程的成绩。定义一个可以存放3个学生的结构体数组，实现对学生2门课平均成绩的排序。

9.3.3　结构体数组的使用

```
#include "stdio.h"
struct student {          // 声明结构体类型，同例9-3
                  long int number;        // 学号
char name[8];             // 姓名
                  float score[2];         // 2门课程的成绩
               };
void sort(struct student stud[],int n);//声明成绩排序函数
void main()
{
struct student stud[3];      //定义结构体数组
      int i,j;                    //输入
      for(i=0;i<3;i++)
      {
scanf("%ld",&stud[i].number);
scanf("%s",stud[i].name);
for(j=0;j<2;j++)
scanf("%f",&stud[i].score[j]);
      }
   sort(stud,3);     //输出
```

```
 printf("\n名次 学号    姓名   数学   英语\n");
      for(i=2;i>=0;i--)
      {printf("%d   ",3-i);
       printf("%ld ",stud[i].number);
       printf("%7s ",stud[i].name);
       for(j=0;j<2;j++)
       printf("%7.1f",stud[i].score[j]);
                  printf("\n");                    // 换行
}}
void sort(struct student stud[],int n)
{    int i,j,min;
    struct student t;
    for(i=0;i<n;i++)
{    min=i;
    for(j=i+1;j<n;j++)
              if((stud[j].score[0]+stud[j].score[1])/2<(stud[min].score[
              0]+stud[min].score[1])/2)
           { t=stud[j];
             stud[j]=stud[min];
             stud[min]=t;
            min=j;}
}}
```

9.3.4　结构体数组指针

下面介绍如何应用结构体指针变量存储结构体数组的首地址，以及如何通过结构体指针变量获得结构体数组的元素及其成员变量。

如所示，首先定义结构体数组student[4]及结构体指针p，然后利用p=student语句将数组的首地址赋值给指针p，使p指向了一维数组student，最后通过p+i即第i个数组元素来逐个输出数组的内容。

9.3.4　结构体数组指针

例**9.5**　利用结构体数组指针输出数组内容。

```
#include "stdio.h"
struct data/*定义结构体类型*/
{
       int year;
       int month;
       int day;
};
struct stu/*定义结构体类型*/
{
    char name[20];
    long num;
    struct data birthday;
};
```

9.3.4 结构体数组指针

```
void main()
{
        int i;
    struct
stu*p,student[4]={{"liying",1,1978,5,23},{"wangping",2,1979,3,14},{"libo",3,198
0,5,6},{"xuyan",4,1980,4,21}};
   /*定义结构体数组并初始化*/
   p=student;/*将数组的首地址赋值给指针p,p指向了一维数组student*/
   printf("姓名，序号，出生年,月,日\n");
   for(i=0;i<4;i++)/*采用指针法输出数组元素的各成员*/
      printf("%s ,%ld ,%d,%d,%d\n",(p+i)->name,(p+i)->num,
          (p+i)->birthday.year,(p+i)->birthday.month,(p+i)->birthday.day);
}
```

9.4 编程举例

例1：简单的数据结构题（普通和指针变量引用两种方法）

```
struct point
{       int x;
       int y;
}
struct point pt;//定义结构体变量
struct point *ppt; //定义结构体指针
ppt=&pt;//将结构体变量pt的地址赋给结构体指针ppt
pt.x=0; /*通过pt访问pt的成员即是普通的结构体引用，即可以通过这
个语句来访问成员x并为其赋值*/
(*ppt).x=0; /*通过ppt访问pt的成员即是指针变量的引用，即可以通过
该语句或下面的等价语句来访问成员x并为其赋值*/
ppt->x=0;
```

9.4 编程举例

例2：结构体数组例题（普通、指针和指针变量引用三种方法）

此处仍使用以上例1已经定义的point结构体。

```
struct point pt_array[3]={{1,0}, {1,1},{0,1}};//定义结构体数组pt_array
struct point *ppt; //定义结构体指针
for(i=0;i<3;i++)/* 普通引用，直接使用各个数组元素即结构体引用成员x
和y */
printf("第%d个结构体元素的x值为：%d，y值为：%d",i,pt_array[i].x,
pt_array[i].y);
```

9.4 编程举例

```
for(i=0;i<3;i++)
printf("第%d个结构体元素的x值为：%d，y值为：%d",i, (pt_array+i)->x,
(pt_array+i)->y); /* 指针引用，使用数组名作为数组起始地址并通过指针
运算逐个引用各元素的成员x和y */
i=0;
for(ppt= pt_array;ppt<pt_array+3;ppt++)
printf("第%d个结构体元素的x值为：%d，y值为：%d",i++,ppt->x, ppt-
>y);
/* 指针变量引用，使用指针变量ppt初始化为数组起始地址并递增来逐个
引用各元素的成员x和y */
```