



Xidian University

C语言程序设计

Lec 9 结构体





引言





复杂数据的表示

- ✚ 假设一个学生信息包括姓名、学号、成绩，将n个学生的信息，按成绩排序后输出
 - 解决方法1：用三个数组分别存放姓名，学号，成绩，对成绩排序，排序中需要交换两个成绩时要同时交换对应的姓名和学号
 - 解决方法2：将每个学生的三个信息作为一个整体，然后根据成绩排序，交换时将学生信息作为整体交换

使用结构体将多个信息组合为一个整体



结构体

❁ 多个信息组合成的一个逻辑整体

- ❑ 每个信息称为结构体的一个成员

- ❑ 每个成员都有类型和名称

- 类型可以是基本类型也可以是数组类型、结构体类型或指针类型，甚至指向当前结构体类型

```
struct student_t{  
    char id[10];  
    char name[20];  
    double score;  
};
```

```
struct complex_t{  
    double real;  
    double image;  
};
```

```
struct node{  
    int value;  
    struct node *next;  
};
```



主要内容

- 9.1 结构体定义与使用
- 9.2 结构体与函数





结构体声明

- ❁ 声明每个成员
- ❁ 声明结构体名称

```
struct student_t{  
    char id[10];  
    char name[20];  
    double score;  
};
```

```
struct complex_t{  
    double real;  
    double image;  
};
```

```
struct point_t{  
    int x;  
    int y;  
};
```





定义结构体变量

❁ 方法1：定义结构体时声明变量

```
//定义两个结构体变量  
struct student_t{  
    char id[10];  
    char name[20];  
    double score;  
} st1,st2;
```

变量只能在定义结构体时定义，使用不便





定义结构体变量

❁ 方法2：在程序其他地方声明变量

```
struct student_t{  
    char id[10];  
    char name[20];  
    double score;  
};
```

```
//定义两个结构体变量  
struct student_t st1,st2;
```

定义结构体变量时必须加上struct关键字





定义结构体变量

- 方法3：先用typedef将结构体定义为一个简单名称，然后用该名称来定义变量

```
typedef struct {  
    char id[10];  
    char name[20];  
    double score;  
} STUDENT;
```

```
//定义两个结构体变量  
STUDENT st1,st2;
```

```
typedef unsigned int UINT;  
  
//定义两个无符号整数变量  
UINT n1,n2;
```





结构体嵌套

- ❁ 结构体定义可以嵌套，但不能引用正在定义的结构体自身（指针方式引用除外）

```
typedef struct {  
    char id[10];  
    char name[20];  
    double score;  
    struct { //嵌套定义  
        int year;  
        int month;  
        int day;  
    } birthday;  
} STUDENT;  
//定义两个结构体变量  
STUDENT st1,st2;
```

```
struct node{  
    int value;  
    struct node next; //错误，不能引用自己  
};
```

```
struct node{  
    int value;  
    struct node *next; //正确，可以用指针方式引用  
};
```





结构体成员访问

- 用圆点运算符(.)可以访问结构体的每个成员

格式：结构体变量. 成员名称

```
typedef struct {  
    char id[10];  
    char name[20];  
    double score;  
} student_t;
```

```
student_t st;  
strcpy( st.id, "03101001");  
strcpy( st.name, "li ming");  
st.score = 90;
```

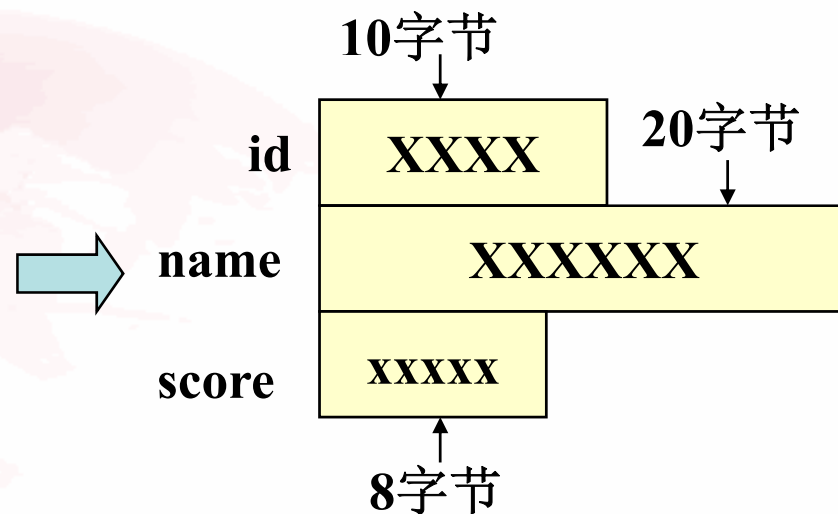


结构体在内存中的存储形式

- 按定义顺序依次存储每个成员

```
//定义两个结构体变量  
struct student_t{  
    char id[10];  
    char name[20];  
    double score;  
} st1,st2;
```

sizeof(st1)=??



sizeof(st1) = 40 ≠ 38

C语言对结构体成员的存储采取了特定的字节对齐方式（1, 2, 4, 8, 16）：

- 各个成员不一定连续存放
- 整个结构体的大小不一定等于各个成员大小之和



结构体在内存中的存储形式

```
struct{  
    char id[10];  
    char name[20];  
    double score;  
} st;
```

```
int main(){  
    printf("sizeof(st)=%d\n", sizeof(st));  
    printf("address of st.id=%d\n", &st.id);  
    printf("address of st.name=%d\n", &st.name);  
    printf("address of st.score=%d\n", &st.score);  
  
    return 0;  
}
```

sizeof(st)=40
address of st.id=4339680
address of
st.name=4339690
address of st.score=4339712





结构体变量初始化

- ❁ **方法1**：定义结构体变量后，逐一初始化每个成员

```
typedef struct {  
    char id[10];  
    char name[20];  
    double score;  
} student_t;
```

```
student_t st;  
  
strcpy(st.id,"03101001");  
strcpy(st.name,"li ming");  
st.score=90;
```



结构体变量初始化

- ❁ **方法2**: 在定义结构体变量时初始化各个成员，各个成员的初始值用逗号分隔，未赋予初值的成员被初始化为0值

```
student_t st={"03101001", "li ming", 90};
```

```
student_t st={"03101001", "li ming"};
```





结构体变量整体赋值

- ❁ 可以用赋值运算符（=）将一个结构体变量的值整体赋值给另一个结构体变量
 - 赋值方式是将一个结构体变量的每个成员依次复制给另一个结构体成员
 - 如果成员是基本类型，按值复制
 - 如果成员是数组，数组内容会被复制
 - 如果成员是指针，复制指针值（地址），导致两个结构体变量的成员指向同一块内存





结构体指针变量

❁ 定义结构体类型的指针变量

■ 格式：结构体类型 *指针变量名

```
typedef struct{  
    int x;  
    int y;  
}point_t;
```

```
point_t pt1={10,10};  
point *pt2=&pt1 ;
```

```
(*pt2).x=20;  
(*pt2).y=20;
```

```
pt2->x=20;  
pt2->y=20;
```



结构体指针变量

引用结构体类型指针变量的成员

- **方法1**：先用间接运算符(*)取得结构体，再用成员运算符(.)访问成员

`(*pt2).x = 20 ;`

此处必须有括号，因为一元运算符是右结合

- **方法2**：直接用专用运算符->取得结构体类型指针变量的成员

`pt2->x = 20 ;`



9.2 结构体与函数





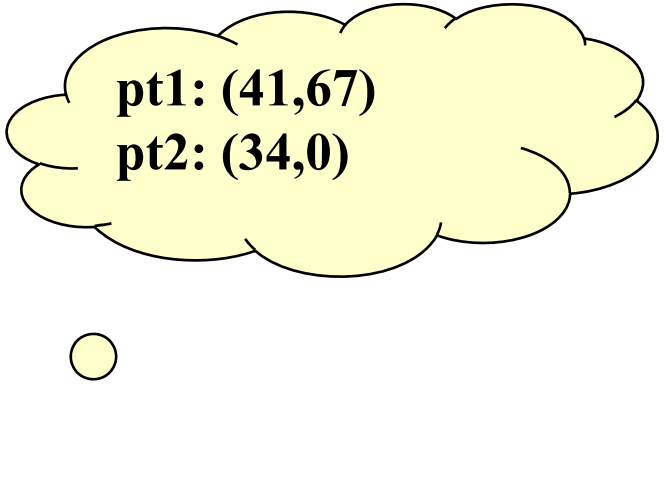
结构体类型作为函数返回值

- ❁ 可以返回一个逻辑整体（相当于返回多个值）
- ❁ 结构体类型返回值可以整体赋值给其他结构体变量



Demo_9_struct_return.c

```
typedef struct{
    int x;
    int y;
}point_t;
point_t random_point(){
    point_t pt;
    pt.x=rand()%100;
    pt.y=rand()%100;
    return pt;
}
int main(){
    point_t pt1,pt2;
    pt1 = random_point();
    pt2 = random_point();
    printf("pt1: (%d,%d)\n", pt1.x, pt1.y);
    printf("pt2: (%d,%d)\n", pt2.x, pt2.y);
    return 0;
}
```



pt1: (41,67)
pt2: (34,0)



结构体类型作为函数形式参数

- ❁ **非指针形式**的结构体类型参数传递时按值传递，即将实参结构体整体赋值给形参，在函数体内对结构体成员的修改不会保留
- ❁ **指针形式**的结构体类型参数传递时，实参和形参指向同一结构体，在函数体内对结构体成员的修改可以保留





结构体类型作为函数形式参数

```
void move1(point_t pt,int dx,int dy){  
    pt.x+=dx;  
    pt.y+=dy;  
}
```

```
void move2(point_t *pt,int dx,int dy){  
    pt->x+=dx;  
    pt->y+=dy;  
}
```

```
int main(){  
    point_t pt1={10,10};  
    printf("pt1: (%d,%d)\n",pt1.x,pt1.y);  
    move1(pt1,5,5);  
    printf("pt1: (%d,%d)\n",pt1.x,pt1.y);  
    move2(&pt1,5,5);  
    printf("pt1: (%d,%d)\n",pt1.x,pt1.y);  
    return 0;  
}
```

pt1: (10,10)
pt1: (10,10)
pt1: (15,15)

```
typedef struct{  
    int x;  
    int y;  
}point_t;
```



示例

- ❁ 假设一个学生信息包括姓名、学号、成绩，原始文件中按学号依次存放了n个学生的信息，要求按成绩排序后输出学生信息

- ❁ step1: 输入学生信息
- ❁ step2: 按成绩排序
- ❁ step3: 输出学生信息

```
typedef struct{  
    char id[10]; //学号  
    char name[20]; //姓名  
    double score; //成绩  
}student_t;
```





step1. 输入学生信息

```
int input( student_t sts[], char *filename ){
    int n=0;
    FILE *fp=fopen(filename,"r");

    if(fp==NULL){
        printf("open input file fail\n");
        return 0;
    }

    while(fscanf(fp,"%s %s %lf",sts[n].id, sts[n].name,&sts[n].score)==3)
        n++;

    fclose(fp);

    return n;
}
```





step2. 按成绩排序

```
void sort( student_t sts[], int n ){  
    int i,j,k;  
    for(i=0; i<n-1; i++){  
        for(k=i,j=i; j<n; j++){  
            if( sts[j].score < sts[k].score) k=j;  
        }  
        if(k!=i){ //交换  
            student_t st=sts[i];  
            sts[i]=sts[k];  
            sts[k]=st;  
        }  
    }  
}
```

选择排序



step3. 输出学生信息

```
int output( student_t sts[],int n,char *filename ){
    int i;
    FILE *fp=fopen(filename,"w");
    if(fp==NULL){
        printf("open input file fail\n");
        return 0;
    }
    for(i=0;i<n;i++){
        fprintf(fp,"%-10s %-20s %.1f\n",
                sts[i].id,
                sts[i].name,
                sts[i].score);
    }
    fclose(fp);
    return 1;
}
```





主函数

```
#define MAX_STUDENT 100

int main(){
    student_t students[MAX_STUDENT];
    int count=0;

    count=input(students, "score.txt");
    sort(students, count);
    output(students,count, "score1.txt");

    return 0;
}
```





小结

- ❊ 结构体的作用
- ❊ 结构体定义
- ❊ 引用结构体成员的方法
- ❊ 结构体与函数
 - 作为函数返回值
 - 作为函数参数

