本小节内容

结构体的定义、初始化、结构体数组
结构体对齐

1 结构体的定义、初始化、结构体数组

有时候需要将不同类型的数据组合为一个整体，以便于引用。例如，一名学生有学号、姓名、性别、年龄、地址等属性，如果针对学生的学号、姓名、年龄等都单独定义一个变量，那么在有多名学生时，变量就难以分清。为此，C 语言提供**结构体**来管理不同类型的数据组合。

声明一个结构体类型的一般形式为

```
struct      结构体名
            {成员表列};
```

例如，

```
struct student
    {
        int num;char name[20];char sex;
        int age;float score;char addr[30];
    };
```

先声明结构体类型，再定义变量名。例如，

```
struct   student    student1, student2;
```

接下来看例 1.1。

【例 1.1】结构体的 scanf 读取和输出。

```
#include <stdio.h>

struct student{
    int num;
    char name[20];
    char sex;
    int age;
    float score;
    char addr[30];
};  //结构体类型声明，注意最后一定要加分号

int main()
{
    struct student s={1001,"lele",'M',20,85.4,"Shenzhen"};  //定义及初始化
    struct student sarr[3];
    int i;
```

```
    printf("%d %s %c %d %f %s\n",s.num,s.name,s.sex,s.age,s.score,s.addr);
    for(i=0;i<3;i++)
    {
        scanf("%d%s %c%d%f%s",&sarr[i].num,sarr[i].name,&sarr[i].sex,&sarr[i].age,
            &sarr[i].score,sarr[i].addr);
    }
    for(i=0;i<3;i++)
    {
        printf("%d %s %c %d %f %s\n",sarr[i].num,sarr[i].name,sarr[i].sex,
            sarr[i].age,sarr[i].score,sarr[i].addr);
    }
    return 0;
}
```

结构体类型声明要放在 main 函数之前，这样 main 函数中才可以使用这个结构体，工作中往往把结构体声明放在头文件中。注意，结构体类型声明最后一定要加分号，否则会编译不通。另外，定义结构体变量时，使用 struct student 来定义，不能只有 struct 或 student，否则也会编译不通，sarr 是结构体数组变量。结构体的初始化只能在一开始定义，如果 struct student s={1001,"lele",'M',20,85.4,"Shenzhen"}已经执行，即 struct student s 已经定义，就不能再执行 s={1001,"lele",'M',20,85.4,"Shenzhen"}。**如果结构体变量已经定义，那么只能对它的每个成员单独赋值，如 s.num=1003。**

采用“结构体变量名.成员名”的形式来访问结构体成员，例如用 s.num 访问学号。在进行打印输出时，必须访问到成员，而且 printf 中的%类型要与各成员匹配。使用 scanf 读取标准输入时，也必须是各成员取地址，然后进行存储，不可以写成&s，即不可以直接对结构体变量取地址。整型数据（%d）、浮点型数据（%f）、字符串型数据（%s）都会忽略空格，但是字符型数据（%c）不会忽略空格，所以如果要读取字符型数据，那么就要在待读取的字符数据与其他数据之间加入空格。

例 1.1 中代码的运行结果如图 1.1 所示，我们可以将运行结果保存到文本文档中。例如，每次运行程序时，在窗口中直接粘贴（通过 ctrl v 来粘贴），进而方便多次测试。

```
1001 lele M 20 85.400002 Shenzhen
1001 lele M 20 85.4 Shenzhen
1005 leli M 21 86.4 Shenzhen
1002 lile M 25 81.4 Shenzhen
1001 lele M 20 85.400002 Shenzhen
1005 leli M 21 86.400002 Shenzhen
1002 lile M 25 81.400002 Shenzhen
```

图 1.1　例 1.1 中代码的运行结果

2 结构体对齐

结构体本身的对齐规则有好几条，比较难记，而且对于考研初试完全没有必要，考研初试只需要记住一条，**结构体的大小必须是其最大成员的整数倍**!

下面我们我们来通过例子来实战

【例】结构体对齐

```
#include <stdio.h>



struct student_type1{

    double score;//double 是一种浮点类型，8 个字节，浮点分为 float 和 double，记住有这两种即可

    short age;

};



struct student_type2{

    double score;

    int height;

    short age;

};



struct student_type3{

    int height;

    char sex;

    short age;

};

int main() {
```

```
    struct student_type1 s1;
    struct student_type2 s2;
    struct student_type3 s3;
    printf("s1 size=%d\n",sizeof(s1));
    printf("s2 size=%d\n",sizeof(s2));
    printf("s3 size=%d\n",sizeof(s3));
    return 0;
}
```

上面例子的运行结果如下：

```
s1 size=16
s2 size=16
s3 size=8
```