第10章 结构体

目录

- 用typedef定义类型
- 定义结构体类型变量的方法
- 结构体变量的引用
- 结构体变量的初始化
- 结构体的使用

用typedef定义类型

- 在已有的数据类型名的基础上用typedef声明新的类型名来代替已有的数据类型。
- typedef定义类型举例

  如：　　typedef　int　INTEGER;

  　　typedef　float　REAL;

  在上述定义的基础上,以下两行等价：

  ① int i，j；　float a，b；

  ② INTEGER　i，j；　REAL　a，b；
- typedef定义类型的功能和目的
  - 适应不同程序员的类型定义习惯
  - 顾名思义
    - 一个整型变量用来计数，可以：typedef int COUNT；

用typedef定义类型

- typedef定义类型的归纳:
  - 声明一个新的类型名的方法是：
    - ① 先按定义变量的方法写出定义体(如：int　i；)。
    - ② 将变量名换成新类型名(如：将i换成COUNT)。
    - ③ 在最前面加typedef(如：typedef　int　COUNT)。
    - ④ 然后可以用新类型名去定义变量。

- typedef定义类型的重要说明：
  - 用typedef可以声明各种类型名，但不能用来定义变量。
  - 用typedef只是对已经存在的类型增加一个类型名，而没有创造新的类型。
  - typedef与#define有相似之处，但也有区别
  - 使用typedef有利于程序的通用与移植。

结构体概述

- 问题：
  - 一个学籍管理系统中，需要学生的学号、姓名、性别、年龄、成绩、家庭地址等信息。这些项都与某一学生相联系。见图。

num	name	sex	age	score	addr
10010	LiFun	M	18	87.5	Beijing

- 处理方法之一：
  - 如果将num、name、sex、age、score、addr分别定义为互相独立的简单变量，缺点是难以反映它们之间的内在联系。
- 处理方法之二：
  - 将这些信息定义在一起

结构体概述

- 结构体类型定义
  - 声明一个结构体类型的一般形式为

```
struct 结构体名
{
   成员表列;
};
```

  - 说明：
    - 注意最后的分号不能省略
    - Struct：声明结构体类型必须使用的关键字，不能省略
    - struct student是一个类型名，它和系统提供的标准类型(如int、char、float、double等)一样具有同样的地位和作用，都可以用来定义变量的类型，只不过结构体类型需要由用户自己指定而已

结构体概述

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

常用方式

```
typedef  struct
{
        int  month;
        int  day;
        int  year;
}DATE;
```

- typedef可以简化结构体类型的声明
  这时就可以用DATE定义变量：
  DATE birthday；（不要写成struct DATE birthday；）
  DATE *p； （p为指向此结构体类型数据的指针）

定义结构体类型变量的方法

- 1.先声明结构体类型再定义变量名
  - 如前面已定义了一个结构体类型struct student，可以用它来定义变量。如:

    struct student　　student1, student2

    StuStudent　　student1, student2

    结构体类型名　　结构体变量名;

student1:10001	Zhang Xin	M	19	90.5	hanghai
student2:10002	Wang Li	F	20	98	Beijing

- 在定义了结构体变量后，系统会为之分配内存单元。例如student1和student2在内存中各占59个字节(2+20+1+2+4+30=59)

定义结构体类型变量的方法

- 关于结构体类型，有几点说明：
  - (1) 类型与变量是不同的概念，不要混同。
    - 只能对变量赋值、存取或运算，而不能对一个类型赋值、存取或运算。
    - 在编译时，对类型是不分配空间的，只对变量分配空间。
  - (2) 对结构体中的成员(即“域”)，可以单独使用，它的作用与地位相当于普通变量。但其引用方式不同于普通变量，关于对成员的引用方法见后面。
  - (3) 成员也可以是一个结构体变量。如：
  - (4) 成员名可以与程序中的变量名相同,二者不代表同一对象

```
struct  date  /*声明一个结构体类型*/
{       int  month;
        int  day;
        int  year;
};
```

```
struct  student
{        int  num;
         char  name[20];
         char  sex;
         int  age;
         struct date birthday;
}  student1, student2;
```

结构体变量的初始化

- 先定义结构体变量，然后对结构体变量的每一个分量进行单独初始化

- 利用赋值运算进行初始化，但进行赋值运算的结构体变量类型必须一致。例如：

```
typedef struct {
        int num;
        char  name[20];
        char  sex;
        int    age;
}STUDENT;
```

```
STUDENT  student1,student2;
student1.num = 1020;
student1.sex = 'F';
student1.age = 20;
strcpy(student1.name, "LiMing");
student2 = student1;
```

结构体变量的使用

- 例如：
  - student1.num=10010;
  - student2．score=student1．score;
  - sum=student1．score+student2．score;
  - student1.age++;
  - ++student1.age;
- 说明
  - "."成员运算符，优先级最高，高于自增自减等单目运算符
  - 不能将一个结构体变量作为一个整体进行输入和输出
  - 例如：
    - cout<<student;　// error

结构体数组

- 定义结构体数组
  - 和定义结构体变量的方法相仿，只需说明其为数组即可。
  - 例如：

    STUDENT stu[3];
- 定义结构体向量
  - vector<STUDENT> vec_student;
  - STUDENT sd1;
  - sd1.num=101010;
  - strcpy(sd1.name, “LiMing”);
  - sd1.sex=‘F’;
  - sd1.age=20;
  - vec_student.push_back(sd1);

```
typedef struct {
        int num;
        char  name[20];
        char  sex;
        int    age;
}STUDENT;
```

结构体类型的指针

- 指向结构体变量的指针
  - 定义的一般形式
    - 结构体类型名 *指针变量名;
- 结构体变量的引用方式
  - 结构体变量.成员名
  - (*p).成员名
  - p->成员名
- 指向运算符->；优先级高于单目运算符
  - p->n :
  - p->n++ ⇔ (p->n)++
  - ++p->n ⇔ ++(p->n)

结构体类型的指针

- 注意事项
  - p++
  - (++p)->num; ⇔ ++p; p->num;
  - (p++)->num; ⇔ p->num; p++;
  - 不允许用指向结构体变量的指针指向结构体变量的成员。如：
    - p = &student1;
    - p = student1.name; //error

结构体变量的举例(CH10_01)

- 假设学生有如下信息：
  - 学号
  - 姓名
  - 性别
  - 成绩：3科课程的成绩

  要求编写程序完成如下操作：
  - 提供学生信息的录入功能
  - 提供学生信息的显示功能
  - 对每个学生的成绩(从高到低)排序的功能