第5章

构造数据类型

C++程序中除基本数据类型 int，float，long，double，char，bool，string 等外，还允许自己声明数据类型。用户声明的数据类型中最为常用的数据类型是“数组”。除数组外，还有结构体类型(structure)、共用体类型(union)、枚举类型(enumeration)、类类型(class)等。这些统称为用户自定义数据类型。

5.1 结构体数据类型

5.1.1 结构体概念

数组是单一数据类型的数据集合。结构体是一个可以包含不同数据类型的一个数据结构，它是一种可以自己定义的数据类型，它和数组主要不同点在于结构体可以在一个结构中声明不同的数据类型，相同结构的结构体变量是可以相互赋值的，而一个数组中的所有数组元素则是相同的数据类型，而且数组名是常量，不能用于整体赋值。结构体相当于其他高级语言或数据库中的记录(record)。

例如，银行储户的基本信息涉及储户账号、姓名、性别、地址、存款日期、存款金额等诸多信息。储户账号、地址是字符串型(string)，存款金额是双精度型(double)，性别是布尔型(bool)，存款日期是日期型(date)。但是，在 C++程序中没有日期型这个基本数据类型，一般通过年、月、日三个整型(int)来替代日期型，通过声明一个名为 date 日期型的结构体数据类型就可以解决这个问题。储户记录由若干不同的数据类型的数据组成，使用数组来处理储户记录需要多个不同的数组，操作过于烦琐，使用结构体数据类型能够方便地处理储户信息的记录数据。

5.1.2 结构体声明

在C++程序中设计一个计算储户定期存款程序需要涉及储户的存款日期、取款日期等有关日期型数据的处理,我们已经知道C++程序中没有像其他语言一样提供“日期/时间”这个基本数据类型,要处理完整的日期型数据,则要声明一个日期型的结构体数据类型。

例如,结构体数据类型日期(date)由年(year)、月(month)、日(day)组成,C++语言允许用户将三个整型组合成一个日期型类型构成结构体,如表5.1所示。

表5.1 结构体date

date		
month	day	year

声明一个名为date的结构体类型。

```
struct date
{int month;
 int day;
 int year;
};
```

声明一个结构体类型date,struct是声明结构体类型的关键字,经过声明后,date就成为一个在本程序中合法使用的数据类型的名称,date结构体类型声明后它和系统提供的int,char,float,double,bool,string等基本数据类型一样可以用来定义变量类型。

声明一个结构体类型的一般形式如下。

```
struct  结构体类型名
{成员表};
```

结构体类型名用来作为结构体类型的标志。上面例子的声明中date是结构体类型名。花括号内是该结构体中的全部成员(member),由它们组成一个特定的结构体。上例中的month,day,year等都是结构体中的成员。在声明一个结构体类型时必须对其成员都进行相应类型声明,结构体成员的数据类型声明格式为“类型名 成员名;”,每一个结构体成员定名规则与变量定名相同。结构体成员可以是常用的数据类型,也可以是已经声明过的结构体类型。它的成员不仅是数据变量(称为数据成员),而且还可以是一个函数(称为成员函数)。C++语言中提供了类(class)类型,一般而言,结构体中不必带有函数成员。

例如,声明两个结构体类型date和depositors。结构体类型date和depositors成员名数据类型如表5.2和表5.3所示。

表5.2 结构体类型date成员名数据类型

成员名说明	成 员 名	数 据 类 型	数据类型说明
月	month	int	整型
日	day	int	整型
年	year	int	整型

表 5.3　结构体类型 depositors 成员名数据类型

成员名说明	成　员　名	数 据 类 型	数据类型说明
储户账号	depositorsid	long	长整型
储户姓名	depositorsname	char [ ]或 string	字符数组型或者字符串型
存款日期	depositdate	date	结构体类型
定期存款额	fixeddeposit	double	双精度型
存款年数	deposityear	int	整型
存款利率	depositrate	double	双精度型
存款利息	depositinterest	double	双精度型

结构体类型 date 及 depositors 声明语句如下。

```
struct date
{int month;
 int day;
 int year;
};

struct depositors
{long depositorsid;
 char depositorsname[40];
 date depositdate;
 double fixeddeposit;
 int deposityear;
 double depositrate;
 double depositinterest;
};
```

在声明 depositors 结构体类型时，其中一个成员 depositdate 是 date 结构体类型，date 类型必须在 depositors 结构体声明前先进行声明。

声明结构体类型的位置一般在文件的开头，在所有函数(包括 main 函数)之前，便于本文件中所有的函数都能利用它来定义变量。当然，也可以在函数中声明结构体类型。

5.1.3　结构体类型变量的定义方法

可以采用下列三种方法定义结构体类型的变量。

1. 方法一

先声明结构体类型，再定义变量的一般形式如下。

```
struct  结构体名
{成员表};
结构体类型名  变量名;
```

例如：

```
date  date1,date2;
```

```
depositors  d1[10];
```

说明：date 和 depositors 是一个结构体类型名，在用结构体类型名定义变量时，date1，date2 是 date 结构体类型的变量，d1[10]是 depositors 结构体类型数组变量。

2. 方法二

在声明类型的同时定义变量的一般形式如下。

```
struct 结构体名
{成员表}  变量名;
```

例如：

```
struct date
{int month;
 int day;
 int year;
}date1,date2;
```

3. 方法三

直接定义结构体类型及变量的一般形式如下。

```
struct
{成员表}变量名;
```

例如：

```
struct
{int month;
 int day;
 int year;
}date1,date2;
```

说明：

方法三虽然合法，但是很少使用。一般提倡使用方法一“先声明结构体类型，再定义变量”，因为将声明结构体类型和定义结构体变量分开，有利于不同的函数甚至不同的文件能够使用所声明的结构体类型。若程序较长，在 C++中允许将若干个结构体类型的声明集中放在一个头文件中，可以使用＃include 指令将头文件包含到本文件中，这个文件就能使用这个结构体类型。

关于结构体类型的几点说明。

(1) 结构体类型的结构根据需要进行设计，可以设计出多种不同的结构体类型。

(2) 类型与变量是两个不同的概念。只能对结构体变量中的成员赋值，而不能对结构体类型赋值。C++在编译时对类型是不分配空间的，只对变量分配空间。

(3) 结构体中的成员可以单独使用，它的作用与地位和普通变量相同。

(4) 结构体成员可以是一个结构体变量。

(5) 结构体中的成员名可以与程序中的变量名相同,二者之间没有关系。

5.1.4 结构体变量的初始化

与其他类型的变量一样,结构体变量可以在定义时进行初始化来指定初值。

例如:

```
struct date
{int month;
 int day;
 int year;
};

struct depositors
{long depositorsid;
 char depositorsname[40];
 date depositdate;
 double fixeddeposit;
 int deposityear;
 double depositrate;
 double depositinterest;
};
depositors e1 = {932111222,"李强",11,30,2013,20000,1,0.0325,650};
```

说明:在 Visual C++ 6.0 中不支持对数据类型为 string 字符串型的结构体成员初始化,所以本例中 depositorsname 采用了字符数组型来实现结构体变量成员 depositorsname 的初始化。

5.1.5 结构体变量的使用

定义了结构体变量就可以使用这个变量及成员,使用结构体变量中成员的一般形式如下。

结构体变量名.成员名

例如:

```
e1.depositorsid        //使用结构体变量 e1 的成员 depositorsid
```

“.”是成员运算符,它在所有运算符中优先级最高,把 e1.depositorsid 作为一个整体来处理。

【例 5-1】 声明两个结构体类型 date 和 depositors,定义一个名为 e1 的结构体变量且数据初始化,输出 e1 变量的各成员数据。

编写程序

```
1  #include <iostream>
2  using namespace std;
3  #include <string>
```

```
4  struct date
5  {int month;
6   int day;
7   int year;
8  };
9  struct depositors
10 {long depositorsid;
11  char depositorsname[40];
12  date depositdate;
13  double fixeddeposit;
14  int deposityear;
15  double depositrate;
16  double depositinterest;
17 };
18 int main()
19 {depositors e1 = {932111222,"李强",11,30,2013,20000,1,0.0325,650};
20  cout <<"depositorsid:"<< e1.depositorsid << endl;
21  cout <<"depositorsname:"<< e1.depositorsname << endl;
22  cout <<"depositdate(month/day/year):"<< e1.depositdate.month <<"/"
23  << e1.depositdate.day <<"/"<< e1.depositdate.year << endl;
24  cout <<"fixeddeposit:"<< e1.fixeddeposit << endl;
25  cout <<"deposityear:"<< e1.deposityear << endl;
26  cout <<"depositrate:"<< e1.depositrate << endl;
27  cout <<"depositinterest:"<< e1.depositinterest << endl;
28  return 0;
29 }
```

运行结果

```
depositorsid:932111222
depositorsname:李强
depositdate(month/day/year):11/30/2013
fixeddeposit:20000
deposityear:1
depositrate:0.0325
depositinterest:650
```

程序分析

(1) 可以将一个结构体变量的值赋给另一个具有相同结构的结构体变量。

例如：

```
depositors  e1 = {932111222,"李强",11,30,2013,20000,1,0.0325,650},e2;
e2 = e1;
```

e1 和 e2 都是 depositors 类型的变量，可以互相整体赋值，赋值时结构体变量 e1 中的各成员的数值分别赋值给结构体变量 e2 中相应的成员。

(2) 可以使用一个结构体变量中的一个成员的值。

例如：

```
e1.depositorsname = "李强";
```

表示结构体变量 e1 中的成员 depositorsname 的值。

(3) 如果成员本身也是一个结构体变量,则必须写到最底一级的成员。

例如:

```
e1.depositdate.month = 11;
```

已定义结构体变量 e1,引用 e1 变量中的 depositdate 成员中的 month 成员,必须逐级引用。e1.depositdate.month,引用结构体变量 e1 中的 depositdate 成员中的 month 成员。

(4) 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:

```
depositors  e1 = {932111222,"李强",11,30,2013,20000,1,0.0325,650};
depositors e2;
```

不能写成:cin >> e2;,也不能写成:cout << e1;。只能对结构体变量中的各成员分别进行输入和输出。

(5) 对结构体变量的成员可以像普通变量一样进行运算。

例如:

```
e1.depositRate += 0.01;
```

(6) 可以引用结构体变量成员的地址,也可以引用结构体变量的地址。

例如:&e1 取 e1 的地址。&e1.depositorsid 取 e1 成员 depositorsid 的地址。

结构体变量的地址主要用作函数参数,将结构体变量的地址作为实参传递给形参。

5.1.6　结构体数组

定义一个结构体数组可以存放一组结构体变量数据。结构体数组与数值型数组不同之处在于每个数组元素都是一个结构体类型的数据,它们又都分别包括各个成员项。

定义结构体数组一般形式如下。

```
结构体类型 数组名[ ] = {初值表};
```

【例 5-2】 定义一个 depositors 结构体类型的变量名为 e 的数组(最多 5 个元素),在定义结构体数组时进行数组元素初始化。执行 print 函数输出数组中 5 个元素的数据。执行 search 函数,通过键盘输入储户账号进行查找,若此账号存在,则将此账号所有存款记录输出;若此账号不存在,则输出"此账号不存在!"。

编写程序

```
1  #include <iostream>
2  using namespace std;
3  #include <string>
4  struct date
```

```
5  {int month;
6   int day;
7   int year;
8  };
9  struct depositors
10 {long depositorsid;
11  char depositorsname[40];
12  date depositdate;
13  double fixeddeposit;
14  int deposityear;
15  double depositrate;
16  double depositinterest;
17 };
18 int main()
19 {depositors e[5] = {
20 {932111222,"李强",11,30,2013,20000,1,0.0325,650},
21 {932111223,"张丽丽",12,21,2013,35000,2,0.0375,2625},
22 {932111654,"伍刚",12,12,2013,17000,5,0.0475,4037.5},
23 {932111987,"徐丽霞",10,4,2013,18500,3,0.0425,2358.75},
24 {932111223,"张丽丽",10,11,2013,9000,2,0.0375,675}};
25  void print(depositors array[]);
26  void search(depositors array[]);
27  print(e);
28  search(e);
29  return 0;
30 }
31 void print(depositors array[])
32 {int i;
33 cout <<"id,name,date(month/day/year),deposit,year,rate,interest"
34 << endl;
35  for(i = 0;i < = 4;i++)
36  { cout << array[i].depositorsid <<","<< array[i].depositorsname <<","
37 << array[i].depositdate.month <<"/"<< array[i].depositdate.day <<"/"
38 << array[i].depositdate.year <<","<< array[i].fixeddeposit <<","<<
39 array[i].deposityear <<","<< array[i].depositrate <<","<<
40 array[i].depositinterest << endl;
41  }
42 }
43 void search(depositors array[])
44 {int i;
45  bool flag  = false;
46  long search_id;
47  cout <<"输入储户账号(depositorsid):";
48  cin >> search_id;
49  for(i = 0;i < = 4;i++)
50  {if (array[i].depositorsid == search_id)
51      { cout << array[i].depositorsid <<","<< array[i].depositorsname <<","
52      << array[i].depositdate.month <<"/"<< array[i].depositdate.day
53 <<"/"<< array[i].depositdate.year <<","
```

```
54      << array[i].fixeddeposit <<","<< array[i].deposityear <<","
55 << array[i].depositrate <<","<< array[i].depositinterest << endl;
56      flag = true;
57    }
58  }
59  if (!flag) cout <<"此账号不存在!"<< endl;
60 }
```

运行结果

```
id,name,date(month/day/year),deposit,year,rate,interest
932111222,李强,11/30/2013,20000,1,0.0325,650
932111223,张丽丽,12/21/2013,35000,2,0.0375,2625
932111654,伍刚,12/12/2013,17000,5,0.0475,4037.5
932111987,徐丽霞,10/4/2013,18500,3,0.0425,2358.75
932111223,张丽丽,10/11/2013,9000,2,0.0375,675
输入储户账号(depositorsid):932111223↙
932111223,张丽丽,12/21/2013,35000,2,0.0375,2625
932111223,张丽丽,10/11/2013,9000,2,0.0375,675

输入储户账号(depositorsid):93211987↙
此账号不存在!
```

程序分析

程序声明了一个全局的结构体类型，在 main 函数中定义了一个局部的结构体数组 e，它有 5 个元素并且进行了数据初始化。通过结构体数组的参数传递，执行 print 函数来输出数组 e 的全部记录。执行 search 函数来实现储户账号的查找和输出。注意：此程序允许一个储户有多笔存款。

5.2　共用体数据类型

5.2.1　共用体类型的声明

共用体类型用来描述类型不相同的数据，与结构体类型不同的是，共用体数据成员存储时采用覆盖技术，共享(部分)存储空间。共用体类型在有的书中亦译为联合体类型。

共用体类型定义用关键字 union 标识，一般形式如下。

union　标识符　{成员表};

例如：声明一个共用体类型，要求包含一个整型成员、一个字符型成员和一个单精度型成员。

```
union  uniondata
    {int    i;
     char   c;
     float  f;
 };
```

共用体变量的定义和结构体变量的定义相类似。定义共用体变量共有三种方法,提倡使用方法一的方式来定义共用体变量。

1. 方法一

先定义共用体类型,再定义共用体变量,一般形式如下。

```
union  共用体名    {成员表};
共用体名 变量表;
```

2. 方法二

定义共用体类型同时定义共用体变量,一般形式如下。

```
union  共用体名    {成员表}  变量表;
```

3. 方法三

直接定义共用体变量,一般形式如下。

```
union    {成员表}    变量表;
```

共用体类型数据占有的存储空间等于占有存储空间最大的共用体成员所占的空间。

5.2.2 共用体类型的举例

【例 5-3】 声明一个结构体来存放学生和教师的数据,含有 id(代码),name(姓名),job(职业)数据,score(成绩)与 jobtitle(职称)共用一段存储空间。使用共用体类型来进行结构的设计,'s'表示学生,'t'表示教师。

编写程序

```
1  #include <iostream>
2  #include <string>
3  using namespace std;
4  struct person
5  {int id;
6     char name[30];
7     char job;
8     union c
9          { int score;
10           char jobtitle[10];
11         }category;
12 };
13 person p[2];
14 int main()
15 {int i;
16       p[0].id=1;
17       strcpy(p[0].name,"王刚");
```

```
18        p[0].job = 's';
19        p[0].category.score = 75;
20        p[1].id = 2;
21        strcpy(p[1].name,"张小强");
22        p[1].job = 't';
23        strcpy(p[1].category.jobtitle,"副教授");
24     for(i = 0;i < = 1;i++)
25     { cout << p[i].id << endl;
26       cout << p[i].name << endl;
27       cout << p[i].job << endl;
28       if (p[i].job == 's') cout << p[i].category.score;
29       else cout << p[i].category.jobtitle;
30       cout << endl;
31     }
32     return 0;
33 }
```

运行结果

```
1
王刚
s
75
2
张小强
t
副教授
```

5.3　枚举数据类型

5.3.1　枚举类型的概念

如果一个变量只有几种可能的值，可以定义为枚举(enumeration)类型。所谓“枚举”是指将变量的值一一列举，变量的值在列举值的范围内。声明枚举类型用 enum 开头。

例如：enum color{blue,yellow,red,black,green};

声明了一个枚举类型 color，大括号中 blue,yellow,red,black,green 等称为枚举元素或枚举常量，表示这个类型的变量的值只能是枚举元素之一，它们是用户自己定义的标识符。

5.3.2　枚举类型的声明

声明枚举类型的一般形式如下。

enum　　枚举类型名　{枚举常量表列};

在声明枚举类型之后，可以用它来定义变量。

例如：color color1,color2；

说明：color1,color2 被定义为枚举类型 color 的变量。根据以上对枚举类型 color 的声明,枚举变量的值只能是 blue,yellow,red,black,green 之一。例如,color1＝blue 是正确的。或直接定义枚举变量：enum{blue,yellow,red,block,green} color1,color2;。

5.3.3 枚举类型的举例

【例 5-4】 从键盘输入 10 个成绩,使用枚举类型来输出对应的等级。成绩与等级对照如下所示。

A_level	85～100
B_level	75～84
C_level	60～74
D_level	0～59

编写程序

```
#include <iostream>
#include <string>
using namespace std;
int main()
{enum grade {A_level,B_level,C_level,D_level};
 grade g1[10];
 int i;float score;
 for (i=1;i<=10;i++)
 {cin>>score;
  switch (int(score/5))
  {case 20:
   case 19:
   case 18:
   case 17:g1[i]=grade(0);break;
   case 16:
   case 15:g1[i]=grade(1);break;
   case 14:
   case 13:
   case 12:g1[i]=grade(2);break;
   default:g1[i]=grade(3);break;
  }
}
  for(i=1;i<=10;i++)
     switch (g1[i])
  {case A_level:cout<<"A_level"<<endl;break;
   case B_level:cout<<"B_level"<<endl;break;
   case C_level:cout<<"C_level"<<endl;break;
   case D_level:cout<<"D_level"<<endl;break;
   default:break;
  }
return 0;
```

```
32  }
```

运行结果

```
输入数据
100↙
85↙
98↙
25↙
74↙
75↙
52↙
65↙
23↙
60↙
输出结果
A_level
A_level
A_level
D_level
C_level
B_level
D_level
C_level
D_level
C_level
```

程序分析

(1) 枚举元素按常量处理,故称枚举常量。枚举元素不是变量,不能对它们赋值,枚举元素的值是固定的。

例如:

```
A_level = 0;          //错误,不能用赋值语句对枚举常量赋值
```

(2) 枚举元素作为常量是有值的,其值是一个整数。编译系统按定义时的顺序对枚举元素赋值为 0,1,2,3,…。在例 5-4 的声明中,A_level 的值为 0, B_level 的值为 1,C_level 的值为 2,D_level 的值为 3。

(3) 枚举值可以用来作判断比较,比较规则按整数进行比较。

例如:

```
if (g1[[0] == A_level])  …
```

(4) 不能把一个整数直接赋给一个枚举变量,枚举变量只能接收枚举类型数据。

例如:

```
g1[0] = (grade)2;
```

或

```
g1[0] = grade(2);
```

不用枚举常量而用常数0代表“A_level”,1代表“B_level”……可以吗?可以。但显然用枚举变量更为直观,枚举元素选用了令人“见名知意”的标识符,而且枚举变量的值限制在定义时规定的几个枚举元素范围内,如果赋予它一个其他的值则会提示出错信息,便于检查。

5.4 typedef声明新的类型名

typedef声明为现有类型创建一个新的名字,或称为类型别名,除了可以用以上方法声明结构体、共用体、枚举类型等外,还可以用typedef声明一个新的类型名来代替已有的类型名。

例如:

```
typedef  float  REAL; //指定用标识符 REAL 代表 float 类型,这样,float i, j;等价于: REAL i,j;
```

也可以对一个结构体类型声明一个新的名字。

例如:

```
typedef struct              //注意在 struct 之前用了关键字 typedef,表示是声明新类型名
{   int num;
    char name[30];
    char sex;
}STUDENT;                   //注意 STUDENT 是新类型名而不是结构体变量名
```

上例声明的新类型STUDENT代表指定的一个结构体类型。习惯上常把用typedef声明的类型名用大写字母表示,以便与系统提供的标准类型标识符相区别。

说明:

(1) 用typedef声明的新类型名称为typedef类型名或typedef名字。用typedef只是对已经存在的类型增加一个类型名而没有创建新的类型。

(2) 用typedef声明新类型名,但不能用来定义变量。例如:typedef int a;是错误的。

(3) 往往会在不同源文件中用到一些类型(尤其是数组、指针、结构体、共用体等类型)时,常用typedef声明这些数据类型,把它们单独放在一个头文件中,只需用#include指令把该头文件包括到本文件中,就可以使用这些typedef类型名,以方便编程,提高编程效率。

(4) 使用typedef类型名,有利于程序的通用与移植。

5.5 综合实例

【例5-5】《选举法》是我国颁布的由广大人民群众选举代表参政议政的法律。选举权和被选举权是公民的基本政治权利之一。选举权是公民选举国家代表机关公职人员的权利。被选举权是公民被选任为国家代表机关公职人员的权利。选举权和被选举权通常

由一国宪法、法律规定并受到保护。我国是一个人民民主专政的社会主义国家，人民当家做主，国家的一切权力属于人民，而保障人民当家做主的方式就是人民代表大会制度，选民在民主选举的基础上产生各级人民代表大会代表，组成地方各级和全国人民代表大会，由他们来代替选民行使职能。

问题描述：现有一个选区，选民采用无记名投票方式选举出席市人民代表大会的人民代表一名。每个选民一人一票，现有四名候选人，候选人姓名分别为“DAI”“LI”“WANG”“ZHAO”，该选区有 n 个选民具有选民资格进行投票，选举规则允许每个选民在四名候选人之外推选出新候选人。投票结束后，按得票数从高到低（降序）输出每个候选人的姓名和得票数，最高得票数者当选为市人民代表。

编程提示

声明一个名为 person 的结构体类型，成员数据类型如下所示。

```
struct person
{   string name;
    int count;
};
```

结构体成员 name 是 string 类型存放候选人姓名，结构体成员 count 是 int 类型存放得票数。

要求：编写程序，在主函数中声明和调用 sort()函数。main()主函数从键盘输入选民数 n，输出且记录 n 个选民所投的候选人姓名，投票结束后按姓名进行计票。调用 sort()函数按得票数进行降序排列，输出每个候选人姓名和得票数，输出最高得票数者当选为该选区的市人民代表。

编写程序

```
1  #include <iostream>
2  #include <string>
3  using namespace std;
4  struct person
5  {   string name;
6      int count;
7  };
8  person leader[100];
9  int num = 4;
10 int main()
11 {   void sort(person x[]);
12     int i,j,n;
13     bool flag = true;
14     leader[0].name = "DAI";leader[0].count = 0;
15     leader[1].name = "LI";leader[1].count = 0;
16     leader[2].name = "WANG"; leader[2].count = 0;
17     leader[3].name = "ZHAO"; leader[3].count = 0;
18     string leadername[100];
19     cout <<"请输入选民人数:";
```

```
    cin >> n;

    for(i = 0;i < n;i++)              // 选民投票
    {cout <<"第"<< i + 1 <<"个选民投票:";
     cin >> leadername[i];
    }

    for(i = 0;i < n;i++)              // 计票
    {flag = true;
        for (j = 0;j < num;j++)
            if (leadername[i] == leader[j].name)
{leader[j].count++;flag = false;}
        if (flag) {leader[num].name = leadername[i];leader[num].count = 1;
 num++;}
      }
      cout <<"按得票数从高到低(降序)输出姓名和得票数:"<< endl;
      sort(leader);                   //排序且输出
      return 0;
}
void sort(person x[])
{int i,j;
 person t;
 for(i = 0;i <= num - 2;i++)
      for(j = 0;j <= num - 2;j++)
          if(x[j].count < x[j + 1].count) {t = x[j];x[j] = x[j + 1];x[j + 1] = t;}
  for(i = 0;i <= num - 1;i++)
      cout <<"姓名"<< x[i].name <<",票数"<< x[i].count << endl;
  cout <<"最高票当选者的是:"<< x[0].name << endl;
}
```

运行结果

(数据仅供参考):

```
请输入选民人数: 10
第 1 个选民投票: LI
第 2 个选民投票: LI
第 3 个选民投票: DAI
第 4 个选民投票: DAI
第 5 个选民投票: LI
第 6 个选民投票: WANG
第 7 个选民投票: LI
第 8 个选民投票: CHENG
第 9 个选民投票: DAI
第 10 个选民投票: LI
按得票数从高到低(降序)输出姓名和得票数:
姓名 LI,票数 5
姓名 DAI,票数 3
姓名 WANG,票数 1
姓名 CHENG,票数 1
```

```
姓名 ZHAO,票数 0
最高票当选者的是: LI
```

【例 5-6】 程序功能,从键盘输入 n 个英文句子(含有空格的字符串),逐行输出英文句子、英文句子中的单词数目及英文句子中所有的单词。说明:单词之间用空格分隔,英文句子不能用空格结束。

编程提示

声明一个名为 line 的结构体类型,成员数据类型如下所示。

```
struct line
{string  sentence;
 int count;
 stringword[100];
};
```

结构体成员 sentence 是 string 类型用于存放英文句子(即含有空格的整个字符串)中所有字符,结构体成员 count 是 int 类型存放一个英文句子中单词的数目,结构体成员 word 数组是 string 类型用于存放英文句子中所有的单词。在主函数中声明和调用函数 input()、filterword()、print()。input()函数从键盘输入 n 个英文句子(含有空格和英文单词)的字符串字符分别放置到 string 数据类型的名为 sentence 成员中;声明和调用 filterword()函数,从每个英文句子字符串中取出所有单词分别放置到 string 数据类型的 word 数组中;声明和调用 print 函数逐行输出每个英文句子、英文句子中的单词数目以及英文句子中所有的单词。

要求:编写程序实现上述功能。

编写程序

```
1  #include <iostream>
2  #include <string>
3  using namespace std;
4  struct line
5  {string sentence;
6   int count;
7   string word[100];
8  };
9
10 line str[100];
11 int n;
12
13 int main()
14 {void input();
15  void print();
16  void filterword();
17  input();
18  filterword();
19  print();
20  return 0;
```

```
21 }
22
23 void input()
24 {cout <<"输入 n:";
25  cin >> n;
26  char x;
27  getchar();
28  int i;
29  for(i = 0;i < = n - 1;i++)
30  {cout <<"输入第"<< i + 1 <<"个英文句子(允许空格的字符串):";
31   while((x = getchar())!= '\n') str[i].sentence = str[i].sentence + x;
32  }
33 }
34
35 void filterword()
36 {int i,j;
37  for(i = 0;i < = n - 1;i++)
38  {bool flag = false;
39  for(j = 0;str[i].sentence[j]!= '\0';j++)
40    if (str[i].sentence[j] == ' ') {if (flag) {str[i].count++;flag = false;}
41                                          else continue;}
42    else
43 {str[i].word[str[i].count] = str[i].word[str[i].count] + str[i].sentence[j];
44 flag = true;}
45  }
46 }
47
48 void print()
49 {int i,j;
50  for(i = 0;i < = n - 1;i++)
51  {cout << str[i].sentence <<"句子中的单词个数是:";
52   cout <<(str[i].count + 1)<<"个,其中的单词是:";
53   for(j = 0;j < = str[i].count;j++) cout << str[i].word[j]<<" ";
54   cout << endl;
55  }
56 }
```

运行结果

(数据仅供参考):

```
输入 n:4
输入第 1 个英文句子(允许空格的字符串):How are you
输入第 2 个英文句子(允许空格的字符串):I am sick
输入第 3 个英文句子(允许空格的字符串):Feel better soon
输入第 4 个英文句子(允许空格的字符串):Thank you
How are you 句子中的单词个数是: 3 个,其中的单词是: How/are/you/
I am sick 句子中的单词个数是: 3 个,其中的单词是: I/am/sick/
Feel better soon 句子中的单词个数是: 3 个,其中的单词是: Feel/better/soon/
Thank you 句子中的单词个数是: 2 个,其中的单词是: Thank/you/
```

本章小结

结构体数据类型是一个可以包含不同数据类型的数据结构，它是一种自己定义的数据类型，它和数组主要不同点在于结构体可以在一个结构中声明不同的数据类型，相同结构体类型的结构体变量是可以相互赋值的；而数组是做不到的。结构体相当于其他高级语言或数据库中的记录(record)。

结构体类型变量的定义方法及初始化。允许使用字符数组型来实现结构体变量成员的初始化，但是，在 Visual C++ 6.0 中不支持对结构体成员数据类型为 string 字符串型的进行初始化。

构造数据类型常用的除数组外，还有结构体类型(structure)、共用体类型(union)、枚举类型(enumeration)、类类型(class)等，这些统称为用户自定义数据类型。

思考题

1. 为什么需要使用结构体类型？
2. 结构体变量的初始化对 string 用 char 数组有什么不同？
3. 能否对结构体类型赋值？
4. 能否对一个结构体变量作为一个整体进行赋值、输入且输出？
5. 结构体变量的赋值与初始化有什么不同？
6. 共用体与结构体有什么区别？

练习题

1. 编写程序，创建一个结构体 ST，其成员有 num(学号)，name(姓名)，score(成绩)，birthday(出生日期)，出生日期是一个名为 date 的结构体类型，初始化第 1 个人员的数据信息并赋予变量 st1，再把此人信息赋给另一个变量 st2，输出变量 st1 及 st2 的值。

2. 编写程序，定义一个结构体 date(包括 year 年、month 月、day 日)，定义变量分别存放出生日期和当日日期。要求输入年、月、日的生日日期和当日日期，并计算出他(她)的实际年龄。

3. 编写程序，有若干学生，每个学生有学号、姓名和成绩等数据。要求：编写一个 inputdata 函数，用于输入 N 个学生的数据。编写一个 outputdata 函数用于输出 N 个学生的数据。编写一个 avgdata 函数用于计算并输出 N 个学生的平均成绩。

4. 编写程序，创建一个结构体 ST，成员有 num(学号)，name(姓名)，score(成绩)，从键盘输入 N 个人员信息，按 score(成绩)降序输出每个学生的记录。在 main 函数中输入数据，在另一个函数中排序并输出。

5. 编写程序，创建一个结构体 PERSON，含有两个成员：id(人员代码)及 name(姓名)。在 main 函数中输入 N 个人员记录数据，编写另一个函数用于从键盘输入人员代

码,到 N 个人员中按 id(人员代码)进行查找,若找到,则显示此人的记录;若找不到,则显示“查无此人!”。

6. 编写程序,创建一个结构体 PERSON,含有两个成员:id(人员代码)及 name(姓名)。在 main 函数中输入 N 个人员记录数据,编写另一个函数用于从键盘输入姓名,到 N 个人员中按 name(姓名)进行查找,若找到,则显示全部同名人员的记录;若找不到,则显示“查无此人!”。