教学质量评价系统登录模块设计

一、项目描述

大部分应用软件都需要注册和登录才能使用，高校的教学质量评价系统也不例外，我们可以利用 C 语言实现教学质量评价系统登录模块设计，要求要对多名教师信息进行初始化，教师信息包括账号、密码和权限 3 个部分，然后输入当前教师的账号、密码和权限进行验证，如果账号和密码均正确，再继续判断权限类别，权限分为“教学督导”和“普通教师”两类。如果账号和密码多次验证都是错误，则显示“账号或者密码错误，登录失败!”，系统流程图如图 1-1 所示。

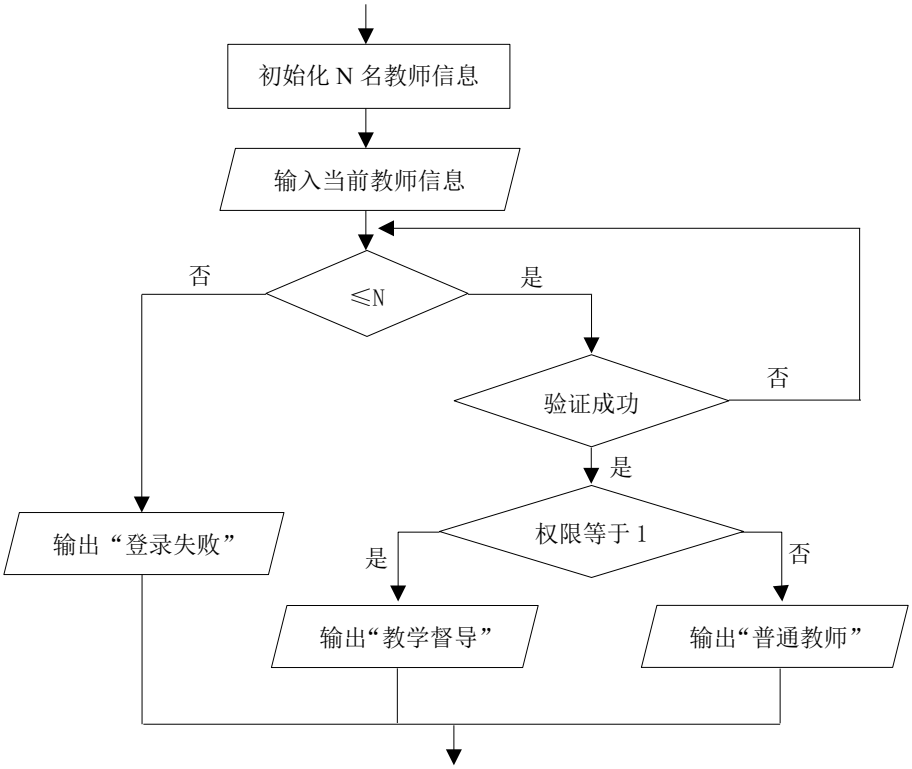


图 1-1 系统流程图

二、项目分析

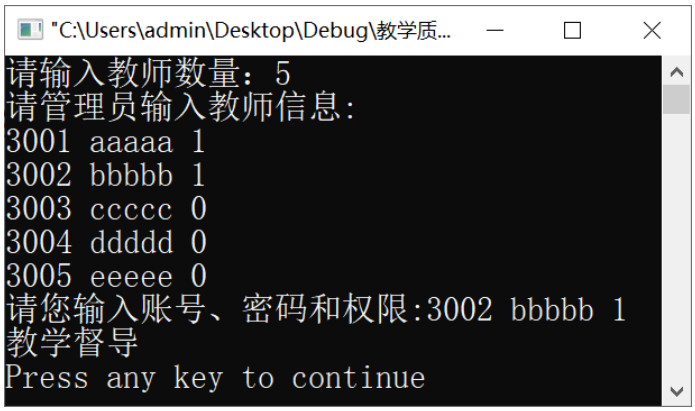
实例中的某位教师所有信息可以称为一个数据元素，由账号、密码和权限 3 个数据项组成。首先，教师信息类型采用结构体类型实现，其中成员包括账号、密码和权限 3 个部分，并且定义一维数组存储多名教师信息，也就是教师信息的初始化，然后，输入当前教师的账号、密码和权限，利用循环结构依次与初始化的教师信息进行比较，比较过程中利用选择嵌套结构进行判断，如果账号和密码均正确，再继续判断权限是否等于 1，如果等于 1，则显示“教学督导”，否则显示“普通教师”，不再继续比较；如果账号和密码多次验证都是错误，则显示“账号或者密码错误，登录失败!”。

三、项目实现

```
#include "stdio.h"
#include "string.h"
//用户结构体定义
typedef struct
{
   long id;
   char pwd[6];
   int pms;
}user;
user users[50];  //数组定义，用来存储多名个教师信息
void init(int n) //输入教师信息
{
    user *p;
    int i;
    p=users;     //指针的应用
    printf("请管理员输入教师信息:\n");
    for(i=0;i<n;i++)
        scanf("%ld%s%d",&p[i].id,p[i].pwd,&p[i].pms);
}
main()
{
   long tid;
   char tpwd[6];
   int tpms;
   int i,n;
   printf("请输入教师数量：");
   scanf("%d",&n);
   init(n);       //调用输入教师信息函数
   printf("请您输入账号、密码和权限:");
   scanf("%ld%s%d",&tid,tpwd,&tpms);
   for(i=0;i<n;i++)
```

```
        {
            if(tid==users[i].id && strcmp(tpwd,users[i].pwd)==0)//验证
            {
                if(tpms==1)
                    printf("教学督导\n");
                else
                    printf("普通教师\n");
                break;
            }
        }
        if(i==n)
           printf("账号或者密码错误,登录失败!\n");
    }
```

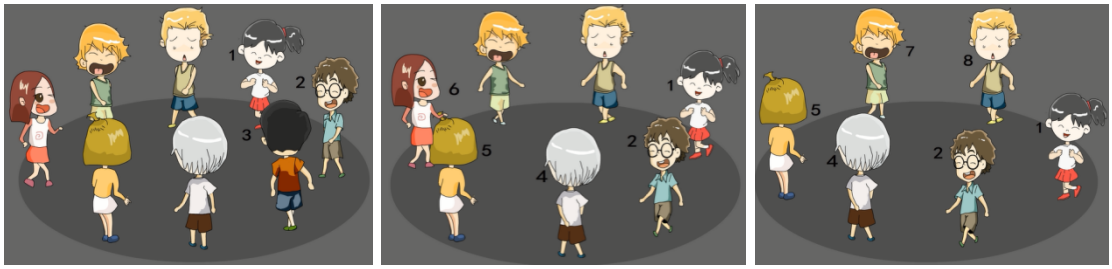
四、结果验证



约瑟夫问题方案设计

一、项目描述

M 个人围成一圈，从第一个人开始依次从 1 循环报数，每当报数为 N 时此人从圈中出来，下一个人又从 1 开始报数，直到圈中只剩下一个人为止。请按退出次序输出出圈人员的编号以及留在圈中的最后一个人原来的编号。实例描述图如图 2-1 所示。



(a) 报数到第 3 个人　　(b) 第 3 个人退出，报数到第 6 个人　　(c) 第 6 个人退出

图 2-1 约瑟夫实例描述图

二、项目分析

M 个人围成一圈，就是将 M 个人的信息存入到一维数组中，当下标达到最大值后重新归零，模拟为循环结构，从某个人（下标：start）开始依次从 1 循环报数，每当报数为 N 时此人从圈中出来，就是将定位到报数为 N（下标：start+N-1）的人删除并输出该人信息，下一个人又从 1 开始报数，报数为 N 的人再次删除并输出该人信息，直到圈中只剩下一个人为止，采用数组存储数据，过程如图 2-2 所示，假设有 10 个人，从第一个人开始报数，报数为 5 的人出圈。

初始化结构：

下标	0	1	2	3	4	5	6	7	8	9
编号	①	2	3	4	5	6	7	8	9	10

第一次出圈的人为 5 号，后面的人前移，从 6 号开始报数：

下标	0	1	2	3	4	5	6	7	8	9
编号	1	2	3	4	⑥	7	8	9	10	

第二次出圈的人为 10 号，从 1 号开始报数：

下标	0	1	2	3	4	5	6	7	8	9
编号	①	2	3	4	6	7	8	9		

第三次出圈的人为 6 号，后面的人前移，从 7 号开始报数：

下标	0	1	2	3	4	5	6	7	8	9
编号	1	2	3	4	⑦	8	9			

第四次出圈的人为 2 号，后面的人前移，从 3 号开始报数：

下标	0	1	2	3	4	5	6	7	8	9
编号	1	③	4	7	8	9				

依次类推，直到最后只剩下一个人为止。

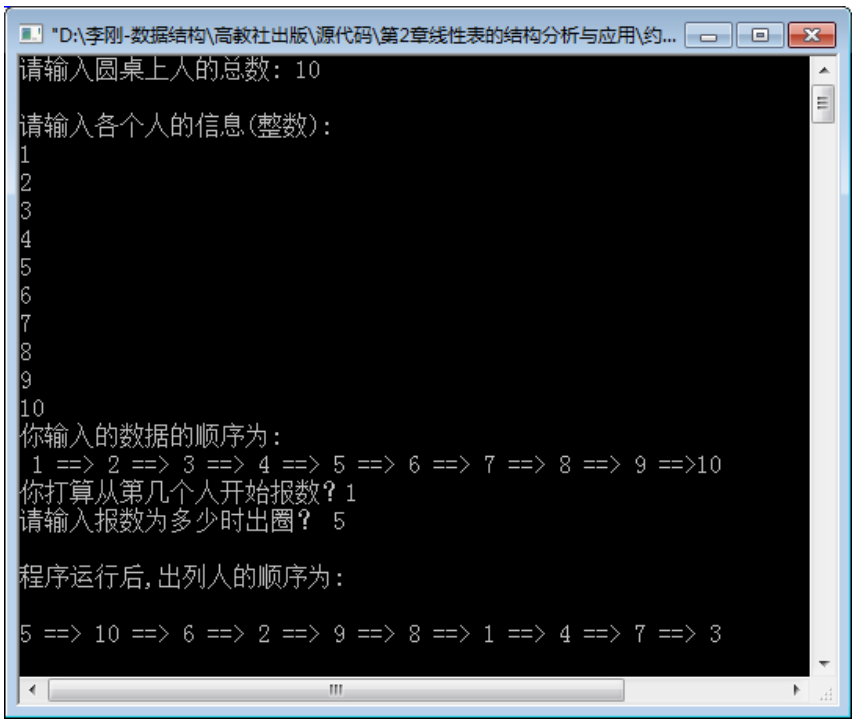
图 2-2 出圈过程描述图

三、项目实现

```
#include "stdio.h"
main()
{
    int person[100];
    int i,j;
    int arrayLen;       //数组长度
    int start, N; //开始位置及报数大小
    int deleNum;        //出列人所在数组中的下标
    int name, M;    //输入时,人的信息以及人的总数
    printf("请输入圆桌上人的总数: ");
    scanf("%d",&arrayLen);
    printf("\n" );
    printf("请输入各个人的信息(整数): \n");
    for(i=0;i<arrayLen;i++)
    {
        scanf("%d",&name);
        person[i]=name;
    }
    printf("你输入的数据的顺序为: \n");
    for(i=0;i<arrayLen-1;i++)
        printf(" %d ==>",person[i]);
    printf("%d\n",person[arrayLen-1]);
    printf("你打算从第几个人开始报数？");
    scanf("%d",&start);
    start=start-1;
    printf("请输入报数为多少时出圈？ ");
    scanf("%d",&N);
    printf("\n");
```

```
        M=arrayLen;
        printf("程序运行后,出列人的顺序为:\n\n" );
        for(i=0;i<M;i++)     //要打印 M 个人的情况,故做 M 次
        {
            if(arrayLen==1)
              printf("%d",person[0]);//数组只剩一个元素,直接出列
            else
            {
              deleNum=(start+N-1)%arrayLen;//此取模保证循环
              printf("%d ==> ",person[deleNum]);
              for(j=deleNum;j<arrayLen;j++)//出列元素后面的各元素前移
                person[j]=person[j+1];
              start=deleNum;
              arrayLen=arrayLen-1;          //移动完毕后,数组长度减 1
            }
        }
        printf("\n\n");
    }
```

四、结果验证



计算器中进制转换功能设计

一、项目描述

计算机中的计算器可以将十进制数转换为二进制数，具体操作过程如图 3-1 所示，例如在图 3-1(a)中输入十进制数 8，然后选择“二进制”单选按钮，如图 3-1(b)所示，界面中显示转换为二进制后的结果 1000，我们要考虑如何实现进制转换的算法设计呢？

(a) (b)

图 3-1 进制转换实例描述图

二、项目分析

十进制数转换为二进制数的方法是循环除以 2 取余数后依次进栈，直到商等于 0 为止，然后将栈中数据再依次出栈得到的序列就是二进制数结果，如图 3-2 所示。

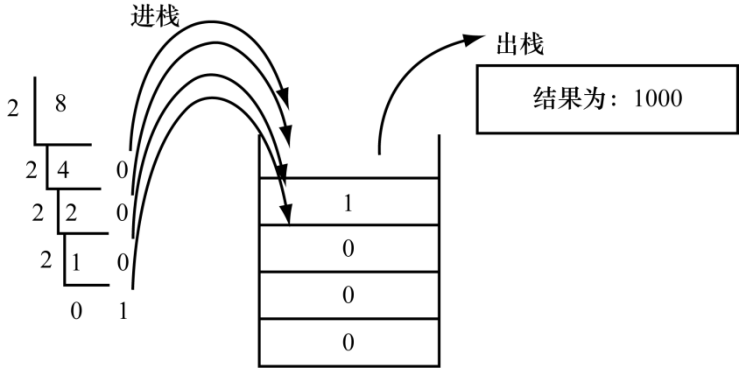


图 3-2 二进制转化的示意图

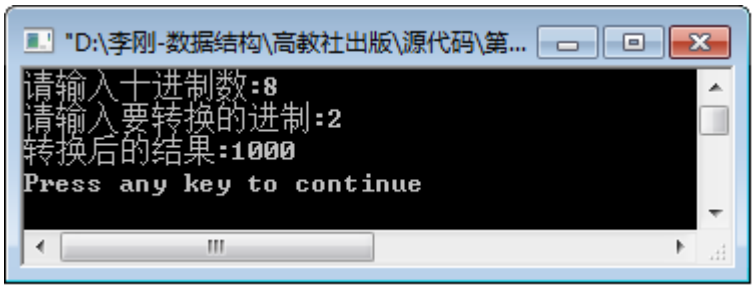
三、项目实现

```
#include "stdio.h"
typedef struct
{
    int data[50];//进制设为整型
    int top;
}seqstack;
//栈的初始化
void initstack(seqstack *s)
{
    s->top=-1;
}
//判断栈是否为空
int empty(seqstack *s)
{
    if(s->top==-1)
        return 1;
    else
        return 0;
}
//进栈操作
void push(seqstack *s,int x)
{
    if(s->top==49)//栈的下标为0-49
        printf("overflow!\n");
    else
    {
        s->top++;
        s->data[s->top]=x;
    }
}
//出栈操作
char pop(seqstack *s)
{
    int x;
```

```
    if(empty(s))
    {
        printf("underflow!\n");
        x='\0';
    }
    else
    {
        x=s->data[s->top];
        s->top--;
    }
    return x;
}
void multibase(int n,int b)
{
    int i;
    seqstack s;
    initstack(&s);
    while(n!=0)
    {
        push(&s,n%b);//进栈
        n=n/b;
    }
    printf("转换后的结果:");
    while(!empty(&s))
    {
        i=pop(&s);//出栈
        printf("%d",i);
    }
    printf("\n");
}
main()
{
    int n,b;
    printf("请输入十进制数:");
```

```
        scanf("%d",&n);
        printf("请输入要转换的进制:");
        scanf("%d",&b);
        multibase(n,b);//调用转换函数
    }
```

四、结果验证



统计一篇英文短文中单词的个数

一、项目描述

我们在阅读英文文章时，为了辨别出每一个单词，会发现英文短文中每个单词都是用空格分开的，现在假设有一篇英文短文，每个单词之间是用空格分开的，试编写一个算法，按照空格数统计短文中单词的个数？例如：图 4-1 所示的一篇英文短文，应该含有 49 个单词。

> To a large degree, the measure of our peace of mind is determined by how much we are able to live on the present moment. Irrespective of what happened yesterday or last year, and what may or may not happen tomorrow, the present moment is where you are always!

图 4-1 英文短文示例

二、项目分析

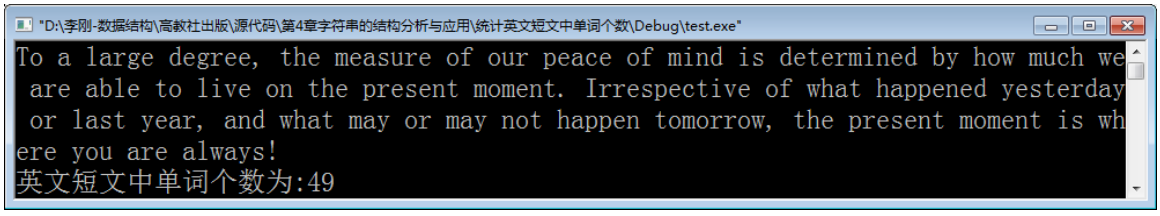
要统计单词的个数先要解决如何判别一个单词，应该从输入行的开头一个字符一个字符地去判别。假定把一篇英文短文放在数组 s 中，那么就相当于从 s[0]开始逐个检查数组元素，经过一个空格或者若干个空格符之后找到的第一个字母就是一个单词的开头，此时利用一个计数器 num 进行累加 1 运算，在此之后若连续读到的是非空格字符，则这些字符属于刚统计到的那个单词，因此不应该将计数器 num 进行累加 1，下一次计数应该是在读到一个空格或者若干个空格符之后再遇到非空格字符开始。因此，统计一个单词时不仅要满足当前所检查的这个字符是非空格，而且要满足所检查的前一个字符是空格。

三、项目实现

```
#include "stdio.h"
#include "string.h"
typedef struct  //字符串结构体定义
{
    char ch[1000];
    int len;
}SeqString;
int numwords(SeqString s)
{
    char prec=' ';
    char nowc;
    int num=0,i;
    for(i=0;i<s.len;i++)
```

```
        {
            nowc=s.ch[i];
            if((nowc!=' ')&&(prec==' ')) //判断当前为非空格，前一个为空格
                num++;
            prec=nowc;
        }
        return num;
    }
    main()
    {
        SeqString s;
        int num;
        char st[1000]={"To a large degree, the measure of our peace of mind
            is determined by how much we are able to live on the present
            moment. Irrespective of what happened yesterday or last year,
            and what may or may not happen tomorrow, the present moment is
            where you are always!"};
        strcpy(s.ch,st);   //将 st 复制给 s
        s.len=strlen(st);  //字符串 st 的长度赋给 s 的 len
        puts(st);          //输出字符串
        num=numwords(s);
        printf("英文短文中单词个数为:%d\n",num);
    }
```

四、结果验证



数据的压缩存储

一、项目描述

已知二维数组 A[5][5]中存放如图 5-1 所示的数据，该二维数组中元素沿着主对角线完全对称，如何能够将数组中的元素进行压缩存储，即只存储下半三角的元素，并且输入数组的行号和列号，可以输出压缩存储后对应元素的值？

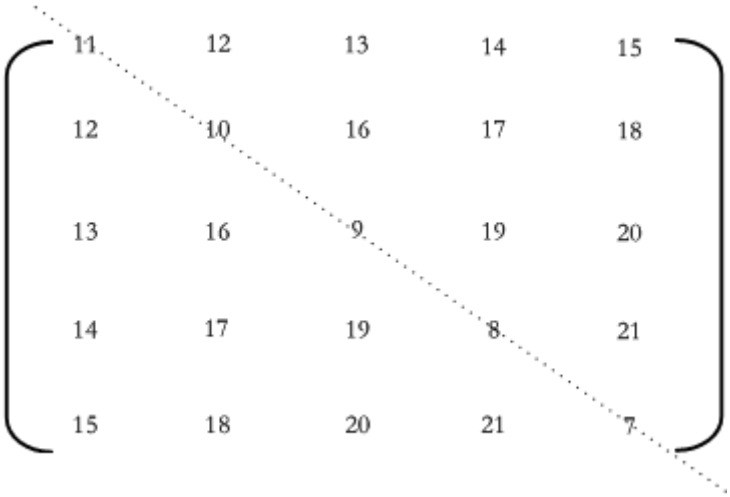


图 5-1 二维数组的存储示意图

二、项目分析

实例描述中图 5-1 所示的二维数组属于对称矩阵，对称矩阵只需要存储下半三角（包括对角线）的元素，在此我们按照行优先存储。具体实现步骤如下：

①定义两个数组，一个二维数组用于存储压缩前对称矩阵中所有元素，另一个一维数组用于存储压缩后对称矩阵中下半三角元素（包括对角线）。

②输入二维数组中所有元素，同时根据行优先存储的对应关系，将下半三角元素存储到一维数组中。

③打印压缩后的一维数组中所有元素和对应位置（下标）。

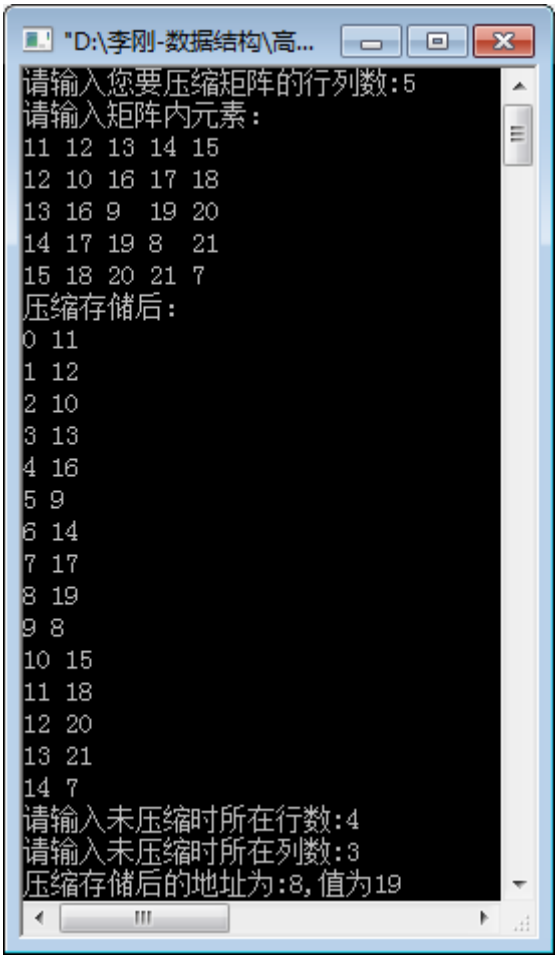
④输入原对称矩阵中的行号和列号，根据行优先存储的对应关系，输出该元素在压缩后一维数组中的位置（下标）。

三、项目实现

```
#include "stdio.h"
void duichen()
{
   int i,j,k,n;
   int L[100][100],SA[100];
   printf("请输入您要压缩矩阵的行列数：");
   scanf("%d",&n);
   printf("请输入矩阵内元素：\n");
   for(i=1;i<n+1;i++)
     for(j=1;j<n+1;j++)
     {
       scanf("%d",&L[i][j]);
       if(i>=j)
         k=i*(i-1)/2+j-1;
       else
         k=j*(j-1)/2+i-1;
       SA[k]=L[i][j];
     }
   printf("您输入的矩阵为：\n");
   for(i=1;i<n+1;i++)
   {
     for(j=1;j<n+1;j++)
       printf("%d ",L[i][j]);
     printf("\n");
   }
   printf("压缩存储后：\n");
   for(k=0;k<n*(n+1)/2;k++)
      printf("%d %d\n",k,SA[k]);
   printf("请输入未压缩时所在行数:\n");
```

```
        scanf("%d",&i);
        printf("请输入未压缩时所在列数:\n");
        scanf("%d",&j);
        if(i>=j)
          k=i*(i-1)/2+j-1;
        else
          k=j*(j-1)/2+i-1;
        printf("该地址的值为：%d\n",SA[k]);
    }
    main()
    {
      duichen();
    }
```

四、结果验证



家族中家谱的设计

一、项目描述

在日常生活当中，家族中经常都有家谱，如图 6-1 所示，该家谱共有四代人，其中李丙方属于最高辈分，李长春、李博宇和李泽宇属于最低辈分，那么我们如何存储家谱中的人员信息和人员之间的关系呢？例如：若输入李泽宇，则输出父亲是李刚，若输入李树林，则输出孩子是李丁香、李贵和李军。

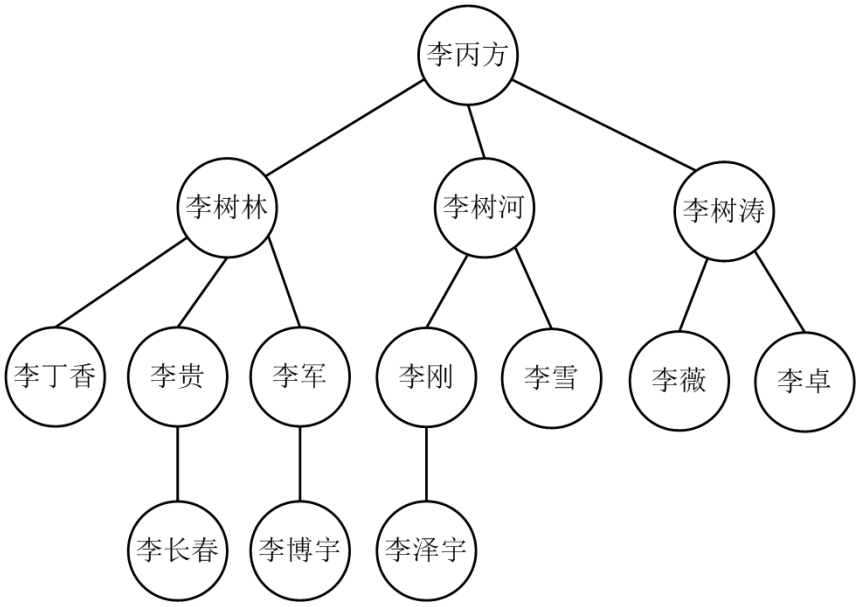


图 6-1 家族家谱图

二、项目分析

家族家谱属于树形结构，我们可以采用双亲表示法存储家谱数据，首先确定家族家谱中的总人数，根据人数定义结构体类型，每个结点应该包含姓名和双亲两个域，创建家谱就是输入姓名和双亲编号，创建家谱后的存储结构如图 6-2 所示。若输出某人的父亲，首先要输入某人姓名，找到此人后记录下双亲编号，再通过双亲编号输出父亲姓名；若输出某人的孩子，首先要输入某人姓名，找到此人后记录下此人对应的下标，再输出双亲等于该下标的孩子姓名。

	0	1	2	3	4	5	6	7	8	9
姓名	李丙方	李树林	李树河	李树涛	李丁香	李贵	李军	李刚	李雪	---
双亲	-1	0	0	0	1	1	1	2	2	---

图 6-2 家族家谱双亲表示法存储结构

三、项目实现

```
#include "stdio.h"
#include "string.h"
typedef struct
{
   char name[8];  //人员姓名
   int parent;    //父亲下标
 }PTreeNode;
typedef struct
{
  PTreeNode nodes[100];
  int n; //人员总数
}PTree;
PTree  people;//全局变量
//创建家谱
void createTree()
{
   int i;
   printf("请输入家谱总人数:");
   scanf("%d",&people.n);
   for(i=0;i<people.n;i++)
   {
     printf("请输入姓名和父亲编号:");
     scanf("%s%d",people.nodes[i].name,&people.nodes[i].parent);
   }
}
//输出某人的父亲
void Parent()
{
   char nam[8];
   int parent,i;
   printf("请输入要查找的人员姓名(查找此人的父亲):");
   getchar();
```

```
        gets(nam);
        for(i=0;i<people.n;i++)
        {
            if(strcmp(people.nodes[i].name,nam)==0)
            {
              parent=people.nodes[i].parent;
              if(parent==-1)
              {
                 printf("此人为最高辈分!\n");
                 break;
              }
              printf("此人的父亲为:%s\n",people.nodes[parent].name);
              break;
            }
        }
        if(i==people.n)
            printf("没有此人!\n");
    }
    //输出某人的孩子
    void Child()
    {
        char nam[8];
        int parent,i,j,tag=0;
        printf("请输入要查找的人员姓名(查找此人的孩子):");
        gets(nam);
        for(i=0;i<people.n;i++)
        {
            if(strcmp(people.nodes[i].name,nam)==0)
            {
              parent=i;
              for(j=i+1;j<people.n;j++)
              {
                if(people.nodes[j].parent==parent)
                {
```