第 1 章 案例研究

1.1 单词检索程序

1. 问题描述

编写一个单词检索程序，通过索引表来实现单词的检索。功能要求如下。

（1）为单词库增加新单词。

（2）通过输入单词查询该单词的所有信息，包括单词拼写、释义和该单词的用例。

（3）建立单词库和索引文件，为后续单词的查找做好准备，将所有单词信息保存到单词库的磁盘文件中。

2. 问题分析与设计

首先要考虑的问题是单词在磁盘文件中的存储及检索方法。建立常驻内存的索引表，索引表的每个索引项由关键字和索引指针组成，其中关键字为单词，索引指针用于保存该单词在单词库中单词记录（结构体）的存放位置，索引表和单词库的结构关系如图 1.1 所示。单词的查询过程是先在索引表找到单词对应的索引项，通过读取索引项的索引指针找到该单词在单词库中的单词结构，然后再读取单词信息。

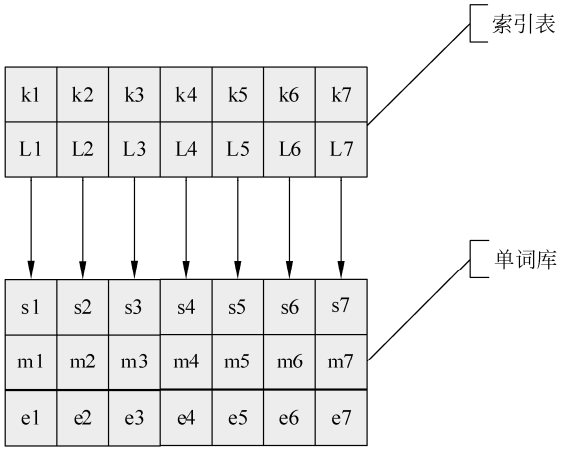

图 1.1

程序启动时从磁盘文件中将索引表和单词库载入内存，程序退出时再保存到磁盘文件中，从而实现持久性存储。

3. 程序实现

```
#include<stdio.h>
#include<string.h>
#include<stdlib.h>
#define MAXI 200
#define WL 30
#define EM 100
//索引表定义
struct IdxType
{
    char key[WL];          //关键字，单词
    long link;             //指向对应块的起始下标，块在文件中
} IDX[MAXI];               //索引表，常驻内存
//单词结构定义
struct Word
{
    char spell[WL];        //英文单词
    char mean[WL];         //中文释义和词性
    char example[WL];      //例子
} WDB[MAXI];               //单词库
long IdxLenght=0;          //索引表长度(单词库中单词数)，随单词的增加会变化
//保存索引表到索引文件
int writeIDX()
{
    FILE *fp;
    if((fp=fopen("IDX.dat","w+"))==NULL)
    {
        printf("打开索引表失败\n");
        return 0;
    }
    rewind(fp);
    fwrite(&IdxLenght,sizeof(int),1,fp);                    //保存索引表长度
    if(IdxLenght>0)
    {
        fwrite(IDX,sizeof(struct IdxType),IdxLenght,fp);//保存索引表
    }
    fclose(fp);
    return 1;
}
//从文件加载索引表到内存
int readIDX()
{
    int n;
    FILE *fp;
```

```
    if((fp=fopen("IDX.dat","r+"))==NULL)
    {
        printf("打开索引表失败\n");
        return 0;
    }
    rewind(fp);
    fread(&n,sizeof(int),1,fp);  //从文件取索引表长度
    if(n>0)
    {
        fread(IDX,sizeof(struct IdxType),n,fp);
    }
    IdxLenght=n;
    fclose(fp);
    return 1;
}
//从文件加载单词库到内存
int readWDB()
{
    FILE *fp;
    if((fp=fopen("WDB.dat","r+"))==NULL)
    {
        printf("打开单词库失败\n");
        return 0;
    }
    rewind(fp);
    if(IdxLenght>0)  //从文件读取单词库
    {
        fread(WDB,sizeof(struct Word),IdxLenght,fp);
    }
    fclose(fp);
    return 1;
}
//保存单词库到文件
int writeWDB()
{
    FILE *fp;
    if((fp=fopen("WDB.dat","w+"))==NULL)
    {
        printf("打开单词库失败\n");
        return 0;
    }
    rewind(fp);
    if(IdxLenght>0)
    {
        fwrite(WDB,sizeof(struct Word),IdxLenght,fp);  //保存单词库
```

```
    }
    fclose(fp);
    return 1;
}
//利用索引表查询单词
long seekWord(char k[])
{
    //在索引表中查找与单词相符的索引项
    int i;
    long pos=-1;
    for(i=0;i<IdxLenght;i++)
    {
        if(!strcmp(IDX[i].key,k))
        {
            pos=IDX[i].link;   //返回单词在单词库的位置
            break;
        }
    }
    return pos;
}
//从单词库读取单词结构
void readWord(long p)
{
    if(p>=0&&p<IdxLenght)
    {
        printf("单词拼写：%s\n",WDB[p].spell);
        printf("单词释义：%s\n",WDB[p].mean);
        printf("例子：%s\n",WDB[p].example);
    }
}
//查询单词
void readRcd()
{
    char k[WL];
    long r;
    fflush(stdin);
    printf("查询单词：");
    gets(k);
    r=seekWord(k);
    if(r==-1)
        printf("未查询到单词\n");
    else
        readWord(r);
}
//增加单词结构到单词库
```

```
int writeWord(long p)
{
    int result=-1;
    struct Word w;
    if(IdxLenght>=MAXI)
        return result;
    else
    {
        fflush(stdin);
        printf("单词拼写：");
        gets(w.spell);
        fflush(stdin);
        printf("单词释义：");
        gets(w.mean);
        fflush(stdin);
        printf("例子：");
        gets(w.example);
        strcpy(IDX[p].key,w.spell);      //在索引表中添加单词索引项
        IDX[p].link=p;
        strcpy(WDB[p].spell,w.spell);    //在单词库中添加单词项
        strcpy(WDB[p].mean,w.mean);
        strcpy(WDB[p].example,w.example);
        IdxLenght++;
    }
    return result;
}
void main()
{
    int s;
    readIDX();
    readWDB();
    do
    {
        printf("1.增加新单词\n2.查询单词\n3.退出\n");
        scanf("%d",&s);
        switch(s)
        {
        case 1:writeWord(IdxLenght);break;
        case 2:readRcd();break;
        case 3:
            if(!writeIDX())
                printf("保存索引表错误\n");
            if(!writeWDB())
                printf("保存单词库错误\n");
            break;
```

```
        default:
            printf("要退出，选3\n");
        }
    }while(s!=3);
}
```

程序的运行结果如图 1.2 所示。

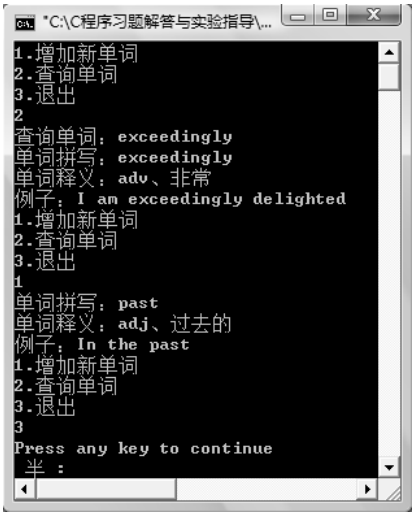

图 1.2

1.2 轮盘游戏程序

1. 问题描述

设计一个 4 人参与的轮盘游戏程序。轮盘上均匀地刻画着 60 条刻度线，并划分为东、南、西、北 4 个方位区，每方位区拥有 15 条连续的刻度。每个参与游戏者可以占据一个方位区。当骰子球（有 6 面，各面分别标有数字 1～6，可以随意自转）落入轮盘后将在轮盘中按顺时针方向转动，同时随意自转。当它经过轮盘上每个刻度时将有一个数字面朝向该刻度，该刻度所属的方位区将获得该数字面所标注的数值分值，等到骰子在轮盘中停止转动时，程序要报出各方位区所获得的积分值，分值高者为获胜方。具体要求如下。

（1）骰子开始落入轮盘时的位置是随机的。

（2）骰子在轮盘中转动圈数是随机的（范围在 100～1000）。

（3）骰子在轮盘中转动的同时也随机转动，经过每个刻度时所朝向的面随机。

2. 问题分析与设计

通过 C 语言的随机函数产生骰子落入轮盘时的位置；骰子在轮盘中转动的圈数；骰子在轮盘中顺时针转动，经过每个刻度时自转的数字；转动停止的位置。骰子在轮盘中转动利用循环语句模拟。

3. 程序实现

```
#include "stdio.h"
```

```
#include "stdlib.h"
#include "time.h"
void win(int *s);
void main()
{
    int score[4]={0,0,0,0},j,pe,end=0;
    long i,n;
    srand(time(NULL));
    n=30000+rand();
    j=rand()%60;
    pe=rand()%60;
    for(i=0;i<n;i++)
    {
        while (1)
        {
            if(j>=60)
            {
                j=0;
                break;
            }
            score[j/15]+=1+rand()%5;
            j++;
            if(j==pe && i==(n-1))
            {
                end=1;
                break;
            }
        }
        if(end)
            break;
    }
    win(score);
}
void win(int *s)
{
    int max=*s,t=0,i;
    char orient[4][3]={"东","南","西","北"};
    printf("各方向得分如下：\n");
    for(i=0;i<=3;i++)
    {
        if(max<*(s+i))
        {
            max=*(s+i);
            t=i;
        }
```

```
        printf("%s方:%d\n",*(orient+i),*(s+i));
    }
    printf("赢家是%s方。\n",*(orient+t));
}
```

程序的运行结果如图 1.3 所示。

图 1.3

1.3 打字练习程序

1. 问题描述

设计一个打字练习程序。程序随机产生并显示一组字符数据供练习者仿照练习输入，练习者输入完毕，按回车键时程序检查核对输入字符与对应位置显示的字符的一致性，同时计算这组输入中正确的字符数；结束练习时程序向练习者显示本次练习的正确率（即正确字符数与显示字符数的比值）。

2. 问题分析与设计

按照练习者设定的长度不断产生字符串，供练习者打字练习，然后将输入字符串和产生字符串进行对比，统计正确率。字符串通过随机函数产生，可以产生分类字符，包括数字、小写字符、大写字符和综合类型字符。

3. 程序实现

```
#include "stdio.h"
#include "stdlib.h"
#include "time.h"
#include "string.h"
#define MAX 100
int total=0,count=0;
void strdisplay(char str[],int n,int st)
{
    int i,b,e;
    switch(st)
    {
    case 1:
        b=48;
        e=57;
        break;
```

```
    case 2:
        b=65;
        e=90;
        break;
    case 3:
        b=97;
        e=122;
        break;
    case 4:
        b=32;
        e=126;
        break;
    }
    for(i=0;i<n;i++)
    {
        str[i]=b+rand()%(e-b);
        printf("%c",str[i]);
    }
    printf("\n");
    str[i]='\0';
    count+=n;
}
void strcount(char str1[],char str2[])
{
    int i,m=0,L1,L2,L,len;
    L1=strlen(str1);
    L2=strlen(str2);
    L=L1<L2?(len=L2-L1,L1):(len=L1-L2,L2);
    for(i=0;i<L;i++)
    {
        if(str1[i]!=str2[i])
            m++;
    }
    total+=m+len;
}
void main()
{
    char str[MAX],str1[MAX]="";
    int len,n=0,st=1;
    printf("你要练习输入的字符数最长不超过：");
    scanf("%d",&len);
    if(len>MAX || len<=0)
        len=MAX-1;
    printf("选择练习的字符范围（1.数字，2.大写字符，3.小写字符，4.全部）：");
    scanf("%d",&st);
```

```
    srand(time(NULL));
    while (1)
    {
        n=1+rand()%len;
        strdisplay(str,n,st);
        fflush(stdin);
        gets(str1);
        if(!strcmp(str1,"End"))
            break;
        strcount(str,str1);
    }
    printf("正确率: %.2f\n",1-(float)total/count);
}
```

程序的运行结果如图 1.4 所示。

图 1.4

1.4 邮件管理程序

1. 问题描述

设计一个邮件管理程序。程序提供 100 个私有信箱（每个信箱拥有唯一的编号），每个信箱可装 10 封信，寄信人按照信箱编号向信箱投递信件(信件至少包含收信人信箱编号、寄信人姓名、信件内容、时间等信息)。收信人随时可以通过输入信箱编号和密码，查看自己的私有信箱中的信件，并可以一次性清空信箱；寄信人则可以按照以上定义的信件内容项，给收信人写信，只要收信人的私有信箱未装满，就可以按其私有信箱编号将信送入收信人的信箱中。

2. 问题分析与设计

通过为邮件和信箱建立结构体类型数组，保存和管理邮件。

3. 程序实现

```
#include "stdio.h"
```