东南大学

C 语言课程设计报告

课程名称： 计算机综合课程设计

学　　院： 土木工程学院

设计题目： 猜单词游戏

级　　别： A 级

学生姓名： 罗保宏

学　　号： 05A13225

同组学生： 刘国安

学　　号： 05A13228

指导教师： 卢瑞华

2014 年 9 月 14 日

目录

1 课程设计任务书

1.1 功能简介

该程序的功能是随机抽出单词本（文件）中的单词，给玩家猜测，并根据猜词情况计算出玩家的得分，最后录入到排行榜并输出文件。每次打开程序时会读入单词本和排行榜文件。

1.2 课程设计要求

单词管理：

程序中用来做谜题的单词必须存放在文本文件中，文件中的单词数不得少于 200 个。

玩家纪录：

程序要求记录前五名成绩。成绩指标自行设计，必须包括时间。记录的时候要求有排名、玩家姓名、成绩三项，并保存在文件中。在程序开始运行的时候就必须读入，以便随时供玩家查询、及时更新成绩。

猜词过程：

1 先请用户输入猜的单词数量，可以有一个默认值。

2 随机抽取单词，对每个单词，系统根据谜底单词长度在屏幕上显示相应个数'-'，假设谜底单词为"hello"，则在屏幕上输出"-----"。

3 玩家输入一个字母进行猜测，如果这个字母不在单词中，系统提示玩家不对；如果猜对字母，比如玩家输入了一个'l'，则在屏幕上输出"--ll-"。

4 重复③，直到玩家在规定的次数内猜出了单词或者超过次数游戏失败。

5 显示玩家每个单词猜对与猜错次数等统计信息。如果玩家猜出单词，计算成绩，如进入前五名提示玩家并记录存储到记录文件中。

6 询问玩家是否开始新一轮猜词，如果玩家选“否”，则系统退到外面的菜单。

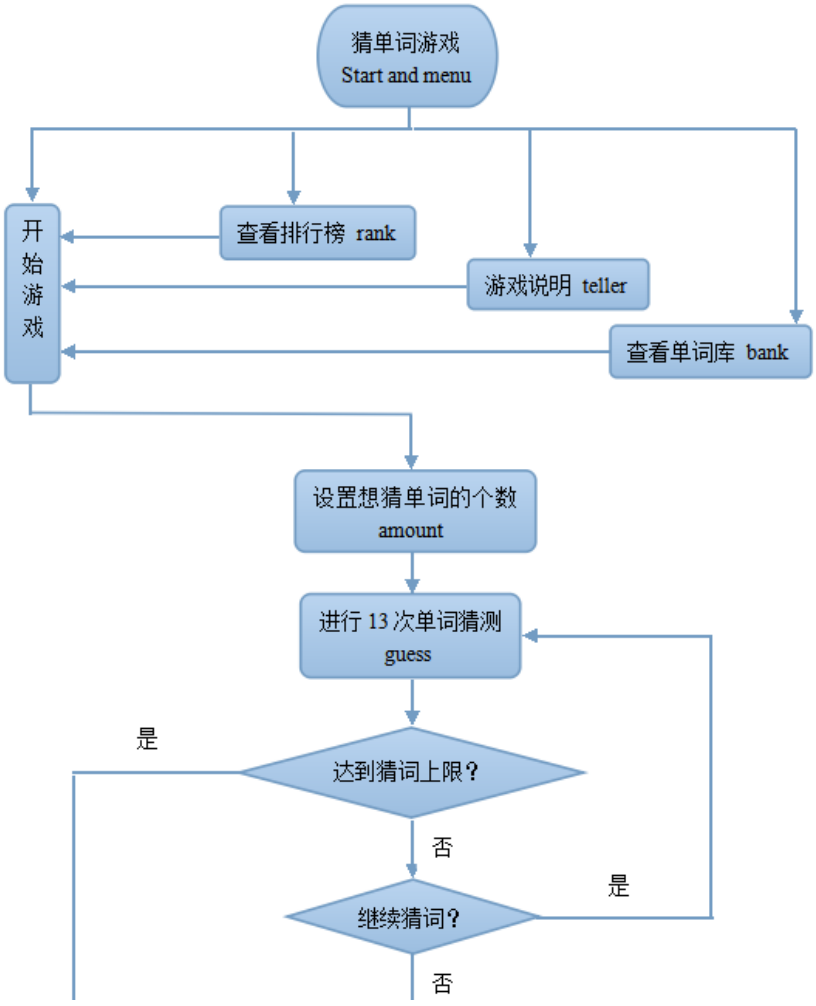
2 系统设计

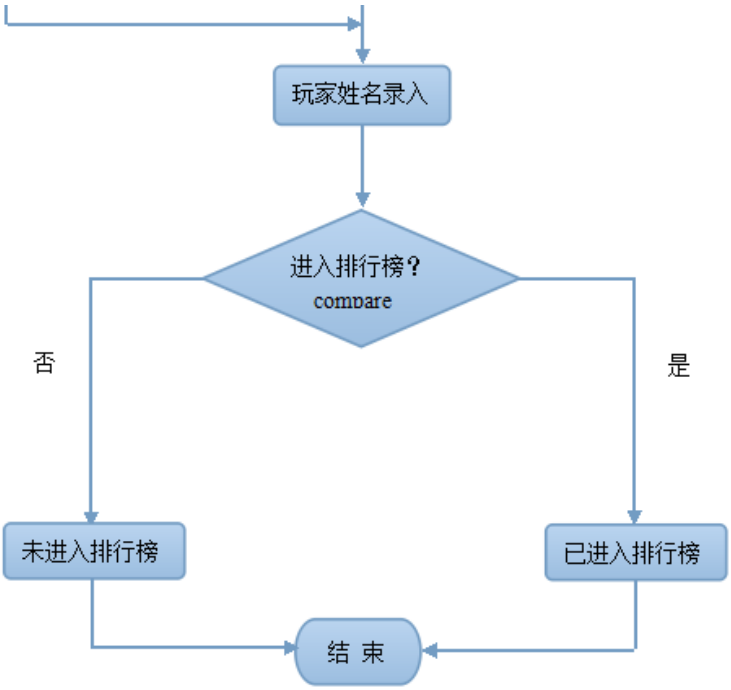
在我们设计的程序中，Main 函数作为一个主线，函数功能块在其中按设计运行顺序依次排列。

```
    void main()
{
    char cho;
    openrank();//读取排行榜文件函数
    start();//欢迎界面
    ffopen();//读取单词本
    menu();//菜单
    amount();//想猜的单词的数量
    int bbc;//大大循环
    .
    .
    .
    compare();
    create();
```

Mian 函数本身要实现的功能很少，主要是通过调用跳转到其他函数实现功能。

程序设计组成框图：




玩家姓名录入
进入排行榜？
compare
否
是
未进入排行榜
已进入排行榜
结 束

3 模块设计

根据 main 函数中的运行过程依次列出模块并解析。

这部分由我们两人分写部分模块并最后合并为整体的“3 模块设计”部分。

3.1 openrank（）读取排行榜文件函数

```
void openrank()
{
    FILE *fp;
    if((fp=fopen("rank.txt","r"))==NULL)
    {
        printf("排行榜不存在\n");
        exit(0);
    }
    int i;
    for(i=0;i<5;i++)
    {
        fscanf(fp,"%s%d",A[i].name,&A[i].grade);
    }
}
```

该函数的作用是读取"rank.txt"文件并把里面的玩家姓名和成绩数据赋给已定义好的全局结构体组的前五个。第六个结构体组用作记录本次玩家姓名和成绩。在 3.8.2 中对结构体有进一步解释。

3.2 start（）欢迎界面函数

```
void start()
{
    printf("    ********************************************    \n");
    printf(" **********          猜单词游戏          ********** \n");
    printf("**********                                  **********\n");
    printf("**********    05A13225        05A13228      **********\n");
    printf(" **********   罗保宏           刘国安     ********** \n");
    printf("    ********************************************    \n");
    printf("\n");
}
```

Start 函数展现的是是整个程序的开头，我们创建友好化界面向屏幕显示游戏名称与作者团队。

3.3 ffopen（）读取单词本函数

```
void ffopen()
{
	FILE *fp;
	if((fp=fopen("words.txt","r"))==NULL)
	{
		printf("单词本不存在\n");
		exit(0);
	}
	int i;
	for(i=0;i<N;i++)
	fgets(words[i],10,fp);//把一行行单词赋给创建好的数组
	//好感动，居然自动从第二行开始读了
	//gets 很特殊，n-1，末位\0 算一个字符
	//fgets 录入最后一行的单词没加\0
}
```

ffopen 函数是为了读取 bank.txt 文件中的单词，供给之后的 bank 函数调用，通过 for 循环和 fgets 指令逐行读取单词，并且 fgets 指令在读取文件中 n 个数据时给前 n-1 个单词在读取的同时末尾自动添加“\0”，最后一个没有添加，导致一旦随机数选中最后一个单词时程序会运行中出现差错，因此我们在单词本中保留 201 个单词，使前 200 个单词正常，并且在赋值给 words[]时限定只有 200 个单词，我们将最后一个单词称为辅助单词 position5。

3.4 menu（）菜单函数

3.4.1 源代码及解析

```
#define R printf("********************按回车键返回菜单********************");getchar();getchar()//简化程序
void menu()
{
	printf("   ************************************************   \n");
	printf("****                    菜单                    ****\n");
	printf("**          1.开始游戏          2.查看排行榜          **\n");
	printf("****        3.游戏说明          4.查看单词库        ****\n");
	printf("   ************************************************   \n");
```

首先创建界面友好化菜单，在屏幕上显示。

```
	int c;
	scanf("%d",&c);
```

随后获取玩家的输入指令，赋给 c。

```
	switch(c)//选择
	{
		case 1:break;
```

```
        case 2:rank();R;menu();break;
        case 3:teller();R;menu();break;
        case 4:bank();R;menu();break;
        default:printf("输入错误，请重新输入\n");menu();break;
    }
}
```

根据玩家的指令 c，利用 switch 条件判断指令，逐个判断，一次指令完成，递归调用 menu 函数，再一次向屏幕上显示菜单栏，随即用 break 指令退出选择判断，防止继续向下进行判断。

C=1，直接跳出执行 guess 函数。

3.4.2 分支——rank（）函数查看排行榜

```
void rank()
{
    printf("************************排行榜************************\n");
    printf("          排名          玩家姓名          成绩          \n");

int i;
for(i=0;i<5;i++)
    {
        if(A[i].grade==-9999)
        printf("          %d          %-14s\b\b\b\b\b\b\b\b\b\b\b\b\b    无          %6d\b\b\b\b\b
无\n",i+1,A[i].name,A[i].grade);
          else
        printf("          %d          %-14s    %6d \n",i+1,A[i].name,A[i].grade);
    }
}
```

C=2，执行 rank 函数查看排行榜。根据 rank.txt 文件中的数据，使用 for 循环逐行输入排行榜中玩家的成绩与姓名。在 rank.txt 文件中如果玩家成绩存在，即为真实成绩；如果排行榜中不存在真实玩家的成绩，即用-9999 代替。在向屏幕中输入成绩时进行判断，如果不等于-9999 则直接输入真实值；如果等于-9999，则利用“\b”隐藏文件中用于占位的玩家姓名 0,与玩家成绩-9999，向屏幕输出“无”。

3.4.3 分支——teller（）函数查看游戏说明

```
void teller()
{
    printf("************************游戏说明************************\n");
    printf("      猜单词游戏的功能是随机抽出单词本（文件）中的单词，\n");
    printf("给玩家猜测，并根据猜词情况计算出玩家的得分，最后录入到\n");
    printf("排行榜中。                                              \n");
    printf("************************计分规则************************\n");
    printf("      每个单词有 13 次机会进行猜测，剩余每次机会得 2 分      \n");
    printf("                  每猜对一个单词得 10 分                  \n");
    printf("  中途退出（没有猜完所选的单词）,每个没猜完的单词扣 20 分\n");
```

```
}
```

C=3，执行 teller 函数，创建程序友好化界面向屏幕输出游戏规则、计分方式等。

3.4.4 分支——bank（）函数查看单词库

```
void bank()
{
    int i;
    for(i=0;i<N;i++)
    printf("%s",words[i]);
    printf("\n");
}
```

C=4，执行 bank 函数。逐行输出由 ffopen 函数读取的单词库，向玩家展示 bank.txt 文件中所有的 200 个单词。

3.4.5 本函数亮点

实现菜单的自我循环——递归调用

递归调用是本函数的一个亮点。之前存在的问题是菜单只能出现一次，不符合使用习惯。想了很多办法想实现菜单的重复出现，都很麻烦。最后突然想起了递归调用这个很不熟悉的方法，竟然很简便地实现了菜单重复。

超长宏定义

以前我们只见过宏定义用数字代替字母，但这次我们尝试了把一长段重复很多次的部分用一个 R 来代替，发现也能行，学会了一项新技能。

```
#define R printf("********************按回车键返回菜单********************");getchar();getchar()
//简化程序
```

3.5 amount（）输入想猜的单词的数量

(前面已定义全局变量 a)

```
void amount()
{
    printf("*****************请输入想猜单词的数量*****************\n"); scanf("%d",&a);
}
```

这个函数最简单，就是读入一个数字，代表想猜单词的数量，并赋给全局变量 a。

3.6 main（）函数内的功能块——大大循环

3.6.1 源代码及其大致分析

```
int bbc;//大大循环
    for(bbc=0;bbc<a;bbc++)
    {
        printf("*************** 正在猜第%2d 个单词…… ****************\n",bbc+1);
        guess();//重要
        printf("已猜出%3d 个单词,当前总得分为%5d\n",rightx,totalgrade);
        if(bbc==a-1) {printf("****************所选单词已全部猜完*****************\n"); break;}
                    printf("**************** 还剩%2d 个单词…… ****************\n",a-(bbc+1));
                    printf("是否开始新一轮猜词?输入 N 结束,输入 Y 继续\n");
        cho=getchar();
        if(cho=='N'||cho=='n') {getchar();totalgrade=totalgrade-(a-bbc-1)*10;
        printf("最终得分为%5d\n",totalgrade);break;}
    }
```

大大循环是最外层的循环，控制着的是猜第一个单词，猜第二个单词，猜第三个单词……猜每个单词内还有几层循环，体现在 guess()函数中。

guess()函数比较庞大，在之后进行解析。

这个功能块中还包含了两次判断。

3.6.2 大大循环内第一个判断—所选单词是否已全部猜完

（前面已进行完这一轮的 guess）

```
if(bbc==a-1) {printf("****************所选单词已全部猜完*****************\n"); break;}
```

其中 a 是全局变量，表示所猜单词的数量。由于循环计数 bbc 从 0 开始，所以当 bbc==a-1 时，第 a 个（即最后一个）单词已被猜过，这时需要表示猜单词结束原因是“所选单词已全部猜完”。

3.6.3 大大循环内第二个判断—是否中途退出（没有猜完所选的单词）

设计这个判断的原因是由于我们设计的计分标准。

```
if(cho=='N'||cho=='n') {getchar();totalgrade=totalgrade-(a-bbc-1)*10;
        printf("最终得分为%5d\n",totalgrade);break;}
```

当玩家中途退出时，由于没有猜完开始所选的单词个数，根据我们设计的计分标准，会受到惩罚，表现就是根据剩余的单词数扣除一定分数。计算语句即为 totalgrade=totalgrade-(a-bbc-1)*10;

3.7 guess（）猜词函数

3.7.1 源程序及其解析

```
void guess()
{
    int r=time();
    int l;
    int i;
    int wrong=0;
    l=strlen(words[r])-1;
    char b[10]={"\0"};
    for(i=0;i<l;i++) b[i]='_';
    for(i=0;i<l;i++) printf("%c ",b[i]);
    printf("\n");
```

Guess 函数的第一部分是为了在单词库中随机为玩家抽选出一个未知单词，并在屏幕上显示出该单词的字母个数。因此用 time 函数取出随机数赋给 r，利用 r 标定所抽选单词的位置，此时可以用 strlen 指令计算出该单词的长短（即字母个数，用“l”表示），随后用 for 循环打印出 l 个空白格，提示玩家单词字母个数。但由于之前 fgets 指令会给单词库中每个单词后面自动添加一个“\0”，所以在用 strlen 计算长度的时候要减去一个单位。

```
    char g;int j;int k;
    int bc;
    for(bc=0;bc<13;bc++)
    {
        k=-100;
        printf("**************你共有 13 次机会，剩余%2d 次***************\n",13-bc);
```

```
getchar();
printf("请输入一个字母进行猜测: ");
g=getchar();

for(j=0;j<l;j++)
if(g==words[r][j]) {k=j;b[k]=g;}

if(k==-100) {wrong++;printf("**********猜错了！这个字母不在单词中**********\n");}
    else                printf("**********猜对了！这个字母在单词中**********\n");

for(i=0;i<l;i++)
printf("%c ",b[i]);printf("\n");
```

Guess 函数的主要作用在于对既定的单词进行循环猜测，在每一次猜测前先告诉玩家共有机会次数与剩余机会次数，利用变量 bc 限定循环次数，同时用总次数 13 减去循环中 bc 的值即为此时玩家的剩余机会次数。将玩家每次的猜测字母赋给“g”，在每次循环中利用 g 与既定单词的每个字母进行比较，由于之前获取玩家想猜单词个数时，玩家在输入个数之后会输入一个“\n”，所以要在获取 g 之前利用 getchar（）吸收掉“\n”。随后需要判断玩家所猜字母正确与否，我们采用位置标定的方法，在每次循环前都初始化 k=-100（位置判断变量），如果玩家猜对字母，就将该字母在单词中所处的位置量 j 赋给 k，再用 g 填补相应空白格，使得在下一次猜测前显示出玩家所猜对的字母，如果猜错 k 值不变，并使 wrong 值加 1（用于统计猜错次数）。之后借助判断位置是否移动（即 k 值是否变化）来断定玩家猜测是否正确。

```
int e;//特有的 end 指标
for(e=0;e<l;e++)
if(b[e]=='_') break;

if(e==l)            {printf("**********                bingo!
**********\n");rightx++;totalgrade=totalgrade+10+(12-bc)*2;break;} //此 if 属于大循环，break 跳出大循环

if(bc==12)  {printf("**********  超 过 次 数 ，  you  lost  **********\n");
totalgrade-=20;                 break;}
```

```
        }
    getchar();
    printf("猜对%d 次,猜错%d 次\n",bc+1-wrong,wrong);
}
```

Guess 函数的最后部分是要最终判断玩家是否完成整个单词的猜测，并显示对错次数。类似于之前 k 值的使用，我们借助 e 变量作为指标。对空白格数组 b[]挨个比较，同时 e 的值随循环逐次增加。如果在比较中任意位置出现空白格则循环提前终止，此时 e 的值小于单词长度 l，即可判断玩家在限定次数内未能猜出整个单词，总分 totalgrade-20；如果最终 e=l，则说明任意位置的空白格都被玩家猜测的字母替代，即成功猜出整个单词，根据我们规定的得分系统计算此时分数。由于在一次猜测后玩家同时又会输入“\n”，故应再次使用 getchar（）。最后在屏幕中显示出猜错的次数 wrong 值，猜对的次数 bc+1-wrong（由于 bc 在猜测循环中初始赋值为 bc=0，故计算猜对次数时需要手动+1）

3.7.2 time（）函数——制造随机数

```
#include <time.h>
int time()
{
    int a;time_t t;
    srand((unsigned)time(&t));
    a=rand()%N;//0<=random<N      取余!!
    return(a);
}
```

由于题目要求是随机抽取一个单词，但我们不知道如何生成随机数，因此生成随机数这一段话我们是直接从网上找的。有些方法生成的是伪随机数，不过这个方法生成的是真随机数，即每次打开程序得到的随机数都不一样。

后来问了卢老师，得知了其中的 srand（） 和 rand（）都是生成随机数，不知怎么搞的最终就能生成一个真随机数。

卢老师还解释了 a=rand()%N 的意思。在这之前，通过反复测试我们知道了 N 就是随机数的上限（而且还是 0<=random<N），但不知道是什么原理。卢老师说%就是取余的意思，生成的数可以随便多大，但是对 N 取余之后绝对不会超过 N，正好符合我们之前的测试情况。

3.8 compare（）比较成绩并调整排行榜函数

3.8.1 源代码

```
void compare()
{
    printf("************猜词结束,敢问大虾尊姓大名？***************\n");
    gets(A[5].name);
    A[5].grade=totalgrade;
    int i,j,k,t;char cname[20];//姓名的中间变量
    for(j=0;j<5;j++)//冒泡法
    for(i=0;i<5;i++)
    if(A[i].grade<A[i+1].grade)
    {
        t=A[i].grade;A[i].grade=A[i+1].grade;A[i+1].grade=t;//换成绩
        for(k=0;k<20;k++)
        {cname[k]=A[i].name[k];A[i].name[k]=A[i+1].name[k];A[i+1].name[k]=cname[k];}//换名字
    }
    if(totalgrade>A[5].grade)  printf("*****************    恭喜进入排行榜!!    *****************\n");//判断是否进入了排行榜
        else          printf("**************    很遗憾,没有进入排行榜!!    **************\n");
}
```

3.8.2 结构体含义

在程序开头我们已定义结构体组

```
struct Rank//建立排行榜姓名-分数的结构体
{
    char name[20];
    int grade;
}A[6];
```

结构体的内容为玩家姓名和成绩。

建立了 6 个结构体，其中前五个结构体装的是排行榜数据，读入由前面的 openrank( )函数首实现。

第六个结构体即在此处的 compare 中接受了玩家输入的姓名（printf("**********猜词结束,敢问大虾尊姓大名？**********\n");gets(A[5].name);）和总成绩（全局变量 totalgrade）。

3.8.3 排行榜构建模式

由 3.8.2 可知以前的 5 个排行榜数据和本次的游戏数据全被装入 6 个结构体中。

之后会采用冒泡法将 6 个结构体中的数据重新降序排序，从而获得了一个新的排行榜。

最后在 create 函数中仅输出前五个数据到新的“rank.txt”中，最后一个分数最低的数据被舍弃。

3.8.4 排序方法——冒泡法

判断指标为成绩的大小，但交换的不只是成绩，还有成绩对应的玩家姓名。

Compare 中有一句 int i,j,k,t;char cname[20];

其中 int i,j 是典型的冒泡法用到的计数变量。

t 是成绩交换的中间变量。t=A[i].grade;A[i].grade=A[i+1].grade;A[i+1].grade=t;

而 k 和 cname[20]都是为交换玩家姓名而定义的。其中 cname[20]是中间变量组。由于已定义好的的字符组不能整体赋值，所以只能再建立一个循环，挨个交换字符。k 即是这个循环中的计数变量。

```
for(k=0;k<20;k++)
{cname[k]=A[i].name[k];A[i].name[k]=A[i+1].name[k];A[i+1].name[k]=cname[k];}
```

3.8.5 判断语句——是否进入排行榜

判断为 if(totalgrade>A[5].grade)，此时结构体组已被重新排序，A[5].grade 是即将被淘汰掉的末位分数，但全局变量 totalgrade 依然没变，是该次猜单词的总成绩。

若该次成绩大于末位分数，说明肯定进入了排行榜。

若该次成绩等于末位分数，根据我的冒泡法设计，相同分数不会被交换，因此末位分数就是该次成绩，即没有进入排行榜（前 5）。

若该次成绩小于末位分数，更不可能进入排行榜。

3.9 create( )输出新 rank.txt 文件函数

```
void create()
{
    FILE *fp;
    if((fp=fopen("rank.txt","w+"))==NULL)
    {
        printf("排行榜不存在\n");
        exit(0);
    }
    int i;
    for(i=0;i<5;i++)
    fprintf(fp,"%s %d\n",A[i].name,A[i].grade);
}
```

这个函数比较简单，就是进行一个标准的“w+”文件操作，“w+”即“为了读和写，建立一个新的文本文件”。

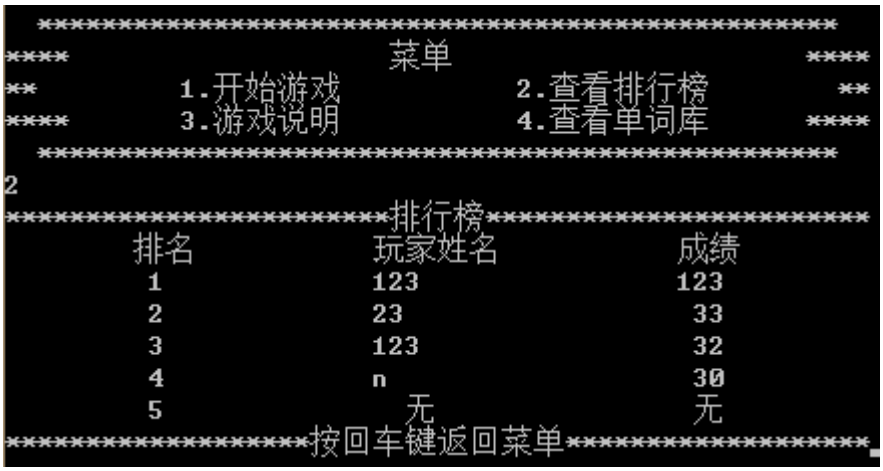
然后再进行向文件输出操作 fprintf，把结构体组的前 5 组数据依次输出到新的“rank.txt”文件中。

这样便建立好了新的“rank.txt”文件，保存下了本次运行程序的结果，供下次运行程序使用。

4 调试与测试

程序运行截屏：

开始



查看排行榜

```
  ****************************************************
*****                    菜单                      *****
**        1.开始游戏           2.查看排行榜          **
****      3.游戏说明           4.查看单词库        ****
  ****************************************************
3
************************游戏说明************************
    猜单词游戏的功能是随机抽出单词本（文件）中的单词，
给玩家猜测，并根据猜词情况计算出玩家的得分，最后录入到
排行榜中。
************************计分规则************************
    每个单词有13次机会进行猜测，剩余每次机会得2分
                每猜对一个单词得10分
 中途退出（没有猜完所选的单词），每个没猜完的单词扣20分
********************按回车键返回菜单********************
```

查看游戏说明

```
  ****************************************************
*****                    菜单                      *****
**        1.开始游戏           2.查看排行榜          **
****      3.游戏说明           4.查看单词库        ****
  ****************************************************
1
******************请输入想猜单词的数量******************
2
**************** 正在猜第 1个单词…… ******************
_ _ _ _
***************你共有13次机会，剩余13次****************
请输入一个字母进行猜测:
*************猜错了！这个字母不在单词中***************
_ _ _ _
***************你共有13次机会，剩余12次****************
```

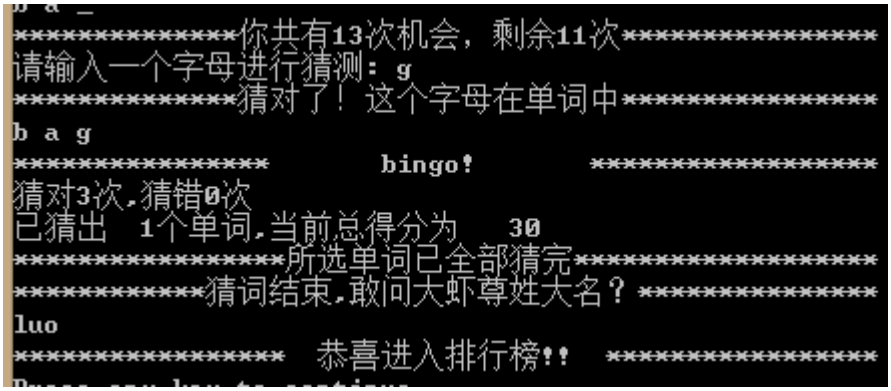
进行猜单词

```
*************猜错了！这个字母不在单词中***************
_ i _ d
**************** 超过次数，you lost ******************
猜对2次，猜错11次
已猜出  0个单词，当前总得分为  -20
****************** 还剩 1个单词…… ********************
是否开始新一轮猜词?输入N结束，输入Y继续
```

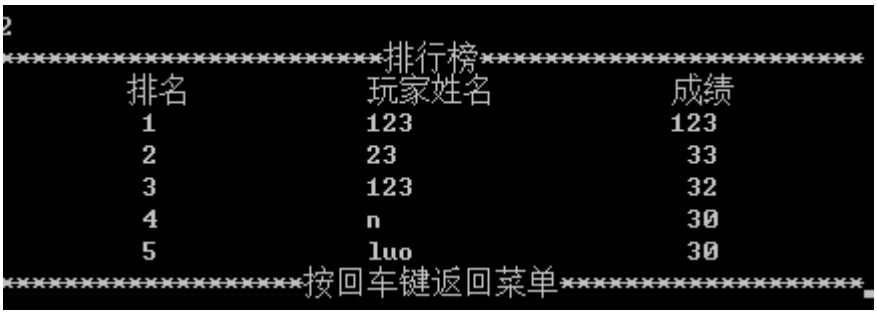
猜错单词

```
是否开始新一轮猜词?输入N结束，输入Y继续
y
**************** 正在猜第 2个单词…… ******************
_ _ _ _ _
***************你共有13次机会，剩余13次****************
请输入一个字母进行猜测: e
```

进行新一轮猜词



猜词成功，计算分数并进入排行榜



进入排行榜展示

5 设计总结

5.1 优点总结

写程序时我们有几个做得很好的地方：

（1） 常用辅助语句，如：

printf("随机数为%d\n",a);//辅助

printf("被取出的为第%d 个单词%s 单词长度为%d\n",r+1,words[r],l);//辅助

利用辅助语句，监测每个关键点的数据是否正确，并辅助程序测试。

（2） 备注和写程序中的感想直接标注在了语句后面

（3） 有清晰的模块意识

5.2 编程中遇到的问题及解决方法

由于前面 60%的程序都是在机房里完成的，前期的一些错误已经没法再展示了。我就举一个后来遇到的问题。

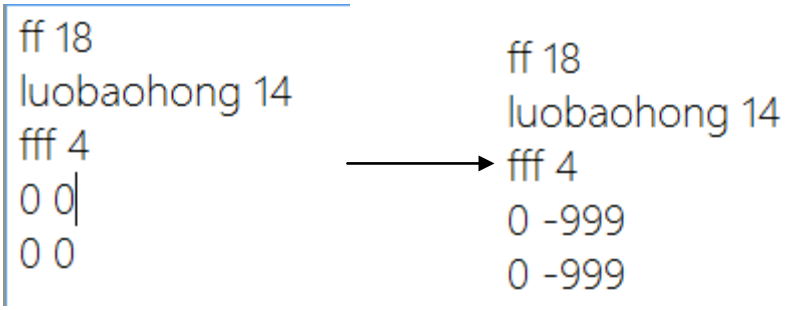
5.2.1 解决排行榜默认值问题

问题描述：

在我们本来设计的排行榜中，我们是把 0（后边那个 0，前面的 0 表示名字）作为一个成绩缺省，意思是这个名次上没人。

但这就有一个问题了，由于我们设计的游戏是存在负分的，要是玩家在这个情况下得了一个负分，按理说缺省值就会被占用。然而我设计的程序是通过比较成绩来排序的，事实上如果得了负分，由于负数小于零，那这个成绩就上不了排行榜！这是一个错误。

于是我想到了把 0 改为-999，一般来说得分不会那么低（除非你故意捣乱），这样怎么比较新的成绩都会比这个缺省成绩大。



但是问题又来了，如果设置为-999.那运行程序后中排行榜显示的就是-999，感觉怪怪的。

于是我就又想了一个办法，

```
if(A[i].grade==-999)
printf("        %d              %-14s %6d\b\b\b\b\b    无\n",i+1,A[i].na
else
printf("        %d              %-14s %6d                \n",i+1,A[i].na
```

由图可知，我采用的办法是用多个\b 退格符退格到-999 之前，并输出空格和无这个字覆盖掉-999，这样

实际上还是-999，但在排行榜这儿看起来就只有文字“无”了。

6 心得体会及致谢

一开始选这个题目，是因为我感觉游戏类的程序会更有趣，写起来也不会那么乏味。正好我的队友刘国安和我的想法是一样的。

事实也是如此，看着设计要求，一个个把问题解决的过程让我和刘国安感到很有成就感。刚开始那几天，我和刘国安两个人每天抽出 3~4 个小时的时间，坐在一起共同讨论，一个字一个字地敲出程序。

很庆幸有队友的互相帮助，我们写程序时常常遇到小问题，但在另一个人的帮助下很快就能解决，每个人都有自己的特长和盲区，我卡住的地方，刘国安常常一眼就能看出问题，这让我们写程序的效率提高很多。想起以前学 C 语言时一个人写程序，遇到卡住的地方老师找不出毛病，很影响效率而且很烦。

在这种情况下，我们的进度很快，每天都有成果。第 4 天后，程序就已经完成了 80%了。

之后因为其他事的原因写程序搁置了很久，但由于基础建立得不错，最后我们还是很轻松地把它完成了。

写这篇报告也给了我很多启发。写报告的时候确实让我对自己的程序的结构有了更清晰的认识。之前程序一直有一个谜一样的问题，偶尔就会出现一次，我们一直搞不懂是怎么回事。本来打算实在没办法就把这个有小瑕疵的程序交了，但是在写这篇报告中的 rank( )函数时，在叙述 rank( )函数作用时，我意外地发现了问题所在，修改并测试后符合我的猜测并解决了问题。

最后很感谢我的队友刘国安和我一起的共同努力，感谢卢老师的答疑帮助。

7 参考文献

《C 语言程序设计（第四版）》　谭浩强著