

**《面对对象程序设计》**

**课程设计**

题目： 教学游戏

院系： 数学与计算科学

专业： 信息与计算科学

学号姓名： 1700710124莫小荷

指导老师： 宁黎华

日期： 2019 年 06 月 30 日

目录

[摘要 2](#_Toc13162316)

[一、课程设计题目及分析 3](#_Toc13162317)

[二、CRC分析过程陈述 4](#_Toc13162318)

[三、UML 类图 6](#_Toc13162319)

[四、主要算法分析 6](#_Toc13162320)

[五、实验过程原始记录（数据、图像、计算等） 18](#_Toc13162321)

[六、心得体会 26](#_Toc13162322)

[七、参考文文献 26](#_Toc13162323)

### 摘要

这次课程设计是编写一个教学游戏，教授学生学习算术计算。程序将对加法、减法和乘法进行测试。游戏共有六个难度级别，第1级测试两个小于10并且和也小于10的数字加法。第2级进行任何两个1位数字的加法；第3级是差为非负的两个1位数的减法；第4级进行任何一个两位数的加法；第5级是差为非负的两个2位数的减法。第6级是行任何一个两位数的乘法。对于每一个问题，游戏参与者都有两次回答问题的机会。在得到5分后游戏进入下一级。为此，本文首先通过建立CRC卡片分析程序要求，拟定要建立的类与功能，然后建立UML类图，把功能函数具体化，最后进行代码的填充和调试。此外，我还设计了一个简单的控制台界面，将游戏参与者分为管理员和游客，管理员需要密码登录，游客可直接进入菜单，使得游戏的互动操作更加友好提高体验用户体验感，同时我还建立了单链表进行对游戏参与者信息的保存，为此还增加了继续游戏、查看历史记录、添加历史记录、同名覆盖、冒泡排序、修改成绩等操作。具体的分析可以详见下文，经过最后运行结果的调试是没有问题，本次课程设计对于我来说也是有一定的启示意义的，让我学到了许多面向对象开发的流程和技巧。

**关键字：教学游戏、算术计算、六个难度级别、管理员和游客、继续游戏、查看历史记录、添加历史记录、同名覆盖、冒泡排序、修改成绩**

## 一、课程设计题目及分析

**1.教学游戏——**编写一个教学游戏，教授学生学习算术计算。程序将对加法和减法进行测试。程序共有三个难度级别，第1级测试两个小于10并且和也小于10的数字加法；第2级进行任何两个1位数字的加法；第3级是差为非负的两个1位数的减法。程序将随机产生问题并读取游戏参与者输入的答案。对于每一个问题，游戏参与者都有两次回答问题的机会。在得到5分后游戏进入下一级。利用CRC卡片设计类和类中的方法，同时给出UML类图并实现该程序。

扩展功能：

**1）**增加第四个难度级别，执行一个两位数的加法

**2)** 能够查询游戏参与者的等级。

**2.分析——**本次课程设计不仅仅需要完成教学游戏的基本内容和基本扩展内容，还要添加额外的可扩展内容使得游戏更加美观易懂切合实际。同时通过建立CRC卡片分析程序要求，拟定要建立的类与功能，然后建立UML类图，把功能函数具体化，最后进行代码的填充和调试。

扩展功能：

**1）**增加第五个难度级别，差为非负的两个2位数的减法。

**2）**增加第六个难度级别，执行一个两位数的乘法。

**3）**增加文件操作，利用单链表形式对文件进行储存和提取相关数据。

**4）**界面设计：将游戏参与者分为管理员和游客，管理员需要密码登录，游客可直接进入菜单。

**5）**以游客进行的游戏参与者可进行的操作有开始游戏、继续上一次的游戏、游戏说明、切换用户。

**6）**以管理员身份进行的游戏参与者可以查看历史记录、创建记录、修改成绩等操作。

**7）**创建记录有两种操作，一种是新建记录，一种是从已有记录的末尾添加记录。

**8）**建立了同名覆盖的操作，在查看历史记录便可对其同名的对象进行覆盖。

**9）**建立排序操作，在查看历史记录时可以利用冒泡排序对文件中的已有信息进行排序。

## 二、CRC分析过程陈述

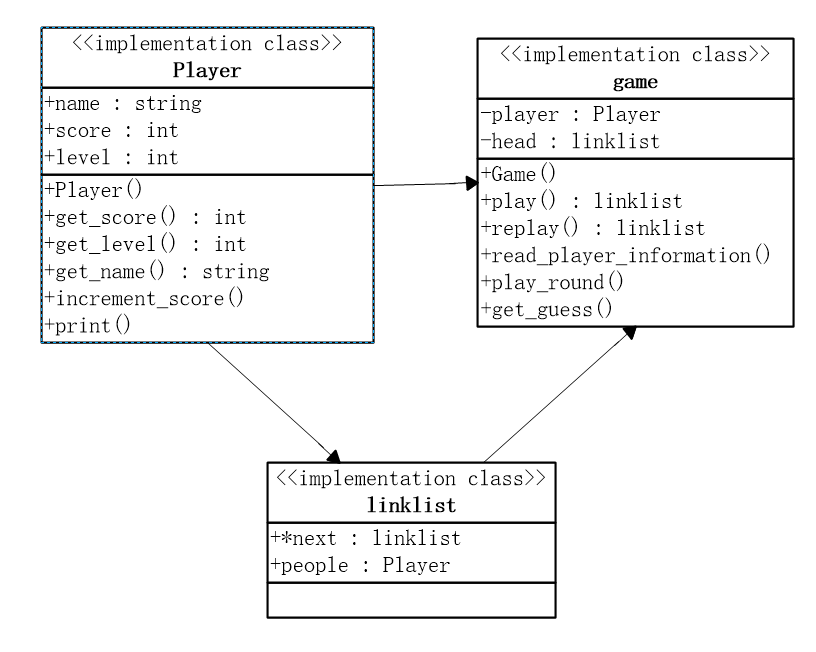
本次设计包含三个类，一个是：Game，一个是：Player，新增linklist类。Game类中包含实现游戏开始，随机数生成，游戏者信息输入和保存等功能的函数，Player类则包含实现游戏者信息返回和分数加成的功能函数，linklist类主要用于文件的操作。由此可得CRC卡片如下：

|  |  |
| --- | --- |
| Class:Player(游戏者类) | |
| 职责： | 协作类 |
| 游戏者初始化值设定  游戏者信息赋值  游戏者得分累加  打印游戏者信息  返回游戏者等级  返回游戏者得分  返回游戏者名称 | linklist(单链表类)  Game（游戏类） |

|  |  |
| --- | --- |
| Class:linklist(单链表类) | |
| 职责： | 协作类 |
| 建立Player(游戏者类)类对象people  建立linklist(单链表类)类对象next | Player(游戏者类)  Game（游戏类） |

|  |  |
| --- | --- |
| Class: Game（游戏类） | |
| 职责： | 协作类 |
| 游戏者初始化  游戏者运行控制  游戏者继续上一次  游戏者信息录入和游戏等级判断  游戏者操作界面控制  游戏记录读取  游戏记录保存 | Player(游戏者类)  linklist(单链表类) |

## 三、UML 类图



## 四、主要算法分析

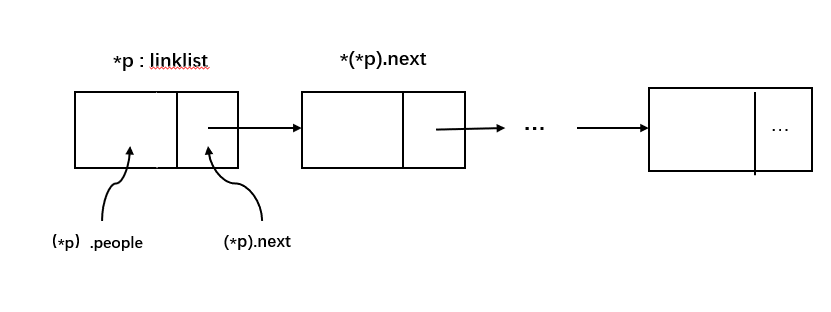
**（1）**为了提高用体验用户体验感，将用户分为管理者和非管理者。用户想成为管理者需要输入密码，如果三次密码输入错误就会自动转到非管理员的界面。主要代码如下：

1. **switch**(choice)
2. {
3. **case** 1:
4. **int** count;
5. **if** (login\_confirm())
6. **return** **true**;
7. **else**
8. {
9. cout<<"    提示：密码错误，你还有两次机会！"<<endl;
10. **for**(count = 1;count<=3;count++)
11. {
12. **if** (login\_confirm())
13. {
14. **return** **true**;
15. **break**;
16. }
17. cout<<"   提示：密码错误！"<<endl;
18. }
19. cout<<"你已经失败三次，自动跳转至普通界面"<<endl;
20. **return** **false**;
21. }
22. **break**;
23. **case** 2:
24. **return** **false**;
25. **break**;
26. **default**:
27. cout<<"没有这个选项！自动为你跳转普通界面"<<endl;
28. **return** **false**;

**（2）**为了更好的与实际情况模拟，玩家成员的储存使用单建链表结构，结构体如下：

1. **class** linklist
2. {
3. **public**:
4. Player people;
5. linklist \*next;
6. };

**（3）**从文件读取玩家信息的数据，将数据存入单链表中。以姓名、成绩、等级为一组的数据形式存入链表类linklist中的people中。示意图：



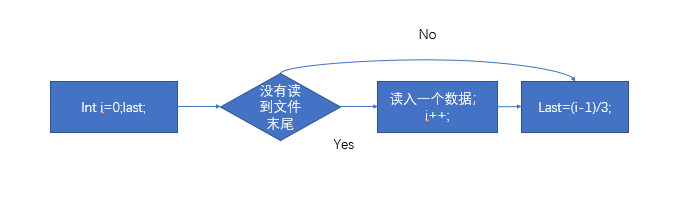
主要代码如下：

1. //从文件中获取数据
2. linklist\* loadFromFile()
3. {
4. ifstream read("family.txt",ios::in);
5. **if**(!read)
6. {
7. cout<<"family.txt文件不存在";
8. system("pause");
9. exit(0);
10. }
12. linklist \*head = NULL;
13. linklist \*rear = NULL;
14. head=rear;
15. **while**(!read.eof())
16. {
17. string nam;
18. **int** grad;
19. **int** scor;
20. read>>nam>>grad>>scor;
21. **if**(nam==" ") **break**;
22. Player new\_person(nam,grad,scor);
23. linklist \*temp = **new** linklist;
24. temp->people = new\_person;
25. temp->next=NULL;
26. **if**(head == NULL)
27. head = temp;
28. **else**
29. rear->next = temp;
30. rear = temp;
31. }
32. rear==NULL;
33. **return** head;
34. }

**（4）查找文件中有几组数据**

算法思想：定义一个i=1和一个last，然后打开文件family.txt读出一个数则i加1直到读完，Last则等于i/3，last为文件的所储存的玩家成员数。

示意图：



主要代码如下：

1. **int** research\_last()
2. {
3. **int** i=0,last;
4. string t;
5. ifstream infile("family.txt",ios::in);
6. **if**(!infile)
7. {
8. cout<<"没有记录"<<endl;
9. system("pause");
10. }
11. **while**(!infile.eof())
12. {
13. infile>>t;
14. i++;
15. }
16. i=i-1;
17. last=i/3;
18. **return** last;
19. }

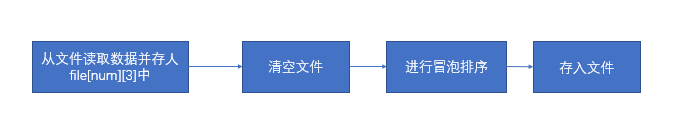
**（5）同名覆盖**

将文件中的相同的名字的低分数的覆盖掉。如果分数相同则将等级低的覆盖掉，算法思想：建立两个定位的指针，两个操作指针，先固定一个不动。另一个往后移动，查找重复元素，查找到进行删除。主要代码如下：

1. linklist \*s,\*p,\*q;  //同名覆盖
2. s=head;
3. p=s->next;
4. q=s;
5. **while**(s)
6. {
7. p=s->next;
8. q=s;
9. **while**(p)
10. {
11. **if**(s->people.name==p->people.name)
12. {
13. **if**(s->people.score<p->people.score)
14. {
15. s->people.level=p->people.level;
16. s->people.score=p->people.score;
17. }
18. **else** **if**(s->people.score==p->people.score&&s->people.level<p->people.level)
19. {
20. s->people.level=p->people.level;
21. s->people.score=p->people.score;
22. }
23. num=num-1;
24. q->next=p->next;
25. free(p);
26. p=NULL;
27. p=q->next;
28. }
29. **else**
30. {
31. q=p;
32. p=p->next;
33. }
34. }
35. s=s->next;
36. }

**（6）用冒泡排序进行玩家的排名**

算法思想：1.先读取文件中的数据并将其存入二维数组，2.清空文件中的内容，3.将二维数组中的信息进行冒泡排序即：重复地走访过要排序的元素列，依次比较两个相邻的元素，如果他们的顺序（如从小到大）错误就把他们交换过来。走访元素的工作是重复地进行直到没有相邻元素需要交换，也就是说该元素列已经排序完成。4.将数据存入文件中。示意图：

主要代码如下：

1. //清空文件的内容
2. **void** fileEmpty(**const** string flieName)
3. {
4. ofstream file(flieName, ios::out);
5. }
6. //排序
7. **void** re\_sort()
8. {
9. **int** num;
10. **int** nume;
11. num=research\_last();
12. string file[num][3];
13. ifstream read("family.txt",ios::in);
14. **for**(**int** i=0;i<num;i++)
15. **for**(**int** j=0;j<3;j++)
16. read>>file[i][j];
18. fileEmpty("family.txt");
19. ofstream output("family.txt", ios::out);
20. string t;
21. **for**(**int** i=0;i<num;i++)
22. **for**(**int** j=0;j<num;j++)
23. **if**(file[i][2]>file[j][2])
24. **for**(**int** k=0;k<3;k++)
25. {
26. t=file[i][k];
27. file[i][k]=file[j][k];
28. file[j][k]=t;
29. }
30. **else** **if**(file[i][2]==file[j][2]&&file[i][1]>file[j][1])
31. **for**(**int** k=0;k<3;k++)
32. {
33. t=file[i][k];
34. file[i][k]=file[j][k];
35. file[j][k]=t;
36. }
38. **for**(**int** i=0;i<num;i++)
39. **for**(**int** j=0;j<3;j++)
40. output<<file[i][j]<<" ";
41. output<<endl;
43. }

**（7）新增记录、增添成员和修改信息时填写的成绩的合理性**

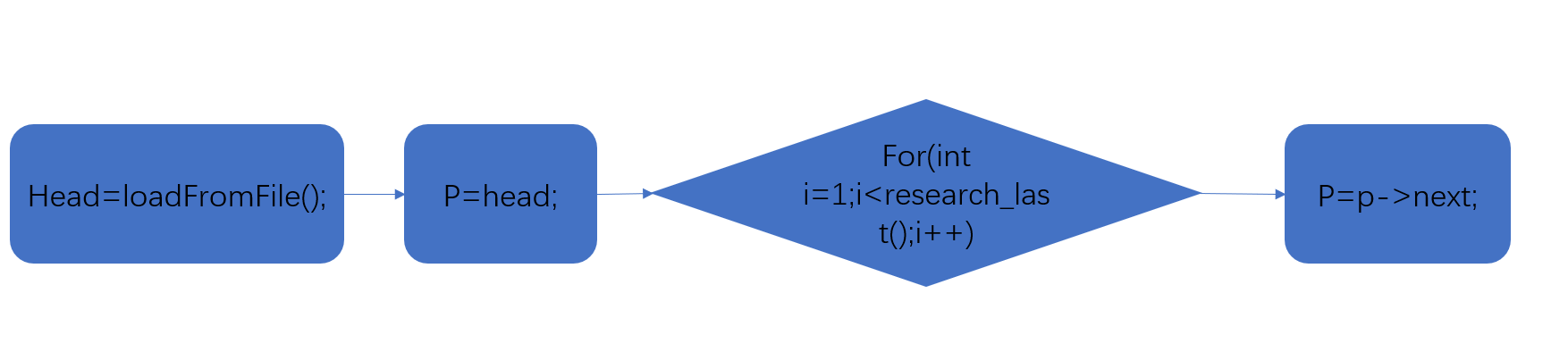
判断成绩合理性算法思想：如果玩家等级为f,则玩家的成绩因大于0且小于f\*5。主要代码如下：

1. //判断数据是否超出范围
2. **int** judge(**int** grad,**int** scor)
3. {
4. **do**{
5. cin>>scor;
6. **if**(scor<1)
7. cout<<"成绩没有负数"<<endl<<"请重新输入：";
8. **else** **if**(scor>grad\*5)
9. cout<<"成绩超出范围" <<endl<<"请重新输入：";
10. }**while**(scor<1||scor>grad\*5);
11. **return** scor;
12. }

**（8）继续游戏和增添成员**

在进行继续游戏和增添成员时的关键是需要寻找链表的最后的节点，算法思想：从文件中读取数据loadFromFile() 并赋值给新建的链表head，将head赋值给新建链表p, 从1开始以文件中获取成员数目research\_last() 为结束不断的进行p=p->next。

示意图：



主要代码如下：

1. head=loadFromFile();
2. linklist \*p=NULL;
3. p=head;
4. **if**(research\_last()==0)
5. {
6. cout<<"你没有历史记录"<<endl;
7. cin>>response;
8. **return** head;
9. }
10. **for**(**int** i=1;i<research\_last();i++)
11. p=p->next;

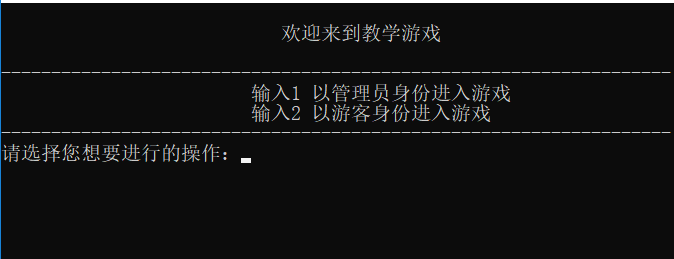
**（9）游戏**

共有六个难度级别，第1级测试两个小于10并且和也小于10的数字加法。第2级进行任何两个1位数字的加法；第3级是差为非负的两个1位数的减法；第4级进行任何一个两位数的加法；第5级是差为非负的两个2位数的减法。第6级是行任何一个两位数的乘法。要实现6个等级和每个级别的功能特性，应该要使用条件语句判定，两个数的生成可用随机函数round（）限定范围实现，在执行了相关的计算和后返回结果；此时再与游戏者输入的答案作比较，一致则输出判定正确语句，否则失败并提示是否再来一次。同时成功之后能显示游戏者的姓名、等级和成绩。

1. //游戏随机数生成及结果返回
2. void Game::play\_round()
3. {
4. int i1 = 0;
5. int i2 = 0;
6. int answer = 0;
7. //设立游戏等级
8. if (player.get\_level() == 1)
9. do
10. {
11. i1 = rand\_int(1,9);  //生成随机数
12. i2 = rand\_int(1,9);
13. answer = i1 + i2;
14. } while (answer > 9);
15. else if (player.get\_level() == 2)
16. {
17. i1 = rand\_int(1,9);
18. i2 = rand\_int(1,9);
19. answer = i1 + i2;
20. }
21. else if (player.get\_level() == 3)
22. do
23. {
24. i1 = rand\_int(1,9);
25. i2 = rand\_int(1,9);
26. answer = i1 - i2;
27. }
28. while (answer < 0);
29. else if (player.get\_level() == 4)
30. {
31. i1 = rand\_int(10,99);
32. i2 = rand\_int(10,99);
33. answer = i1 + i2;
34. }
35. else if (player.get\_level() == 5)
36. do
37. {
38. i1 = rand\_int(10,99);
39. i2 = rand\_int(10,99);
40. answer = i1 - i2;
41. }
42. while (answer < 0);
43. else if (player.get\_level() == 6)
44. {
45. i1 = rand\_int(10,99);
46. i2 = rand\_int(10,99);
47. answer = i1 \* i2;
48. }
49. //提示参与者达到的等级
50. int guess;
51. if (player.get\_level()==1||player.get\_level()==2||player.get\_level()==4)
52. guess = get\_guess(i1, i2, answer, "sum");
53. else if(player.get\_level()==3||player.get\_level()==5)
54. guess = get\_guess(i1, i2, answer, "minus");
55. else
56. guess = get\_guess(i1, i2, answer, "multiply");
57. //答对提示
58. if (answer == guess)
59. {
60. player.increment\_score();
61. cout << "Congratulations, " << player.get\_name() << "! That is correct." << "\n";
62. cout<<"your current level is "<<player.get\_level()<<endl;
63. }
64. else  //答错提醒
65. {
66. cout << "Sorry, " << player.get\_name() << ". The correct answer is "
67. << answer << ".\n";
68. }
69. cout << "Your score is " << player.get\_score() << "\n";
70. }

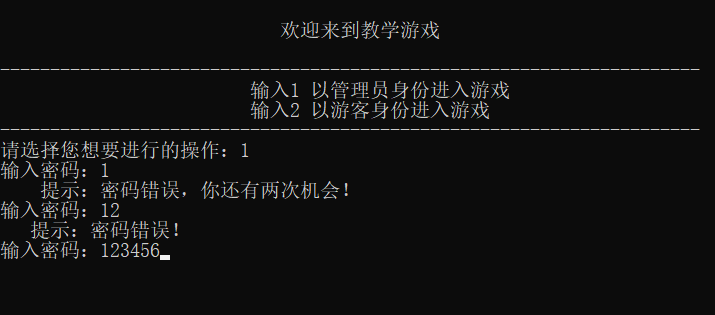
## 五、实验过程原始记录（数据、图像、计算等）

**1.首页菜单**

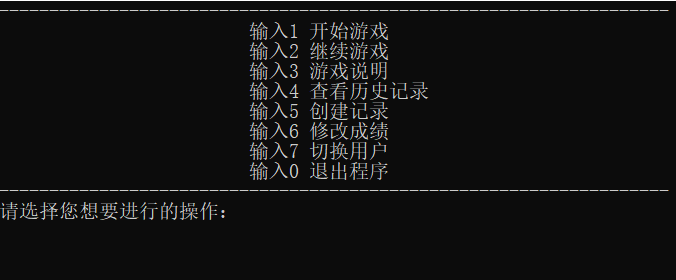


**2.以管理员身份进入菜单**

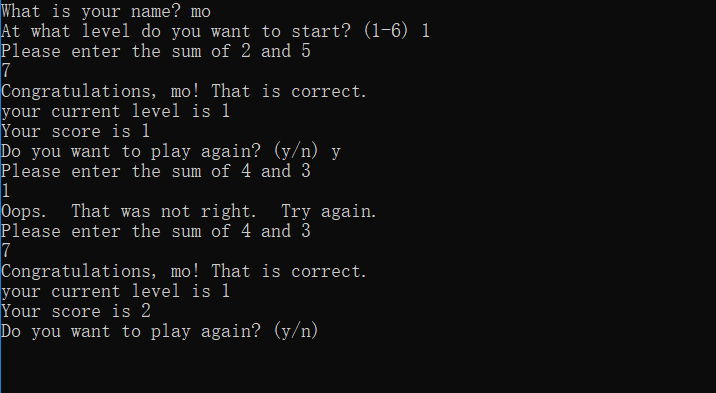
（1）进行密码操作有三次机会



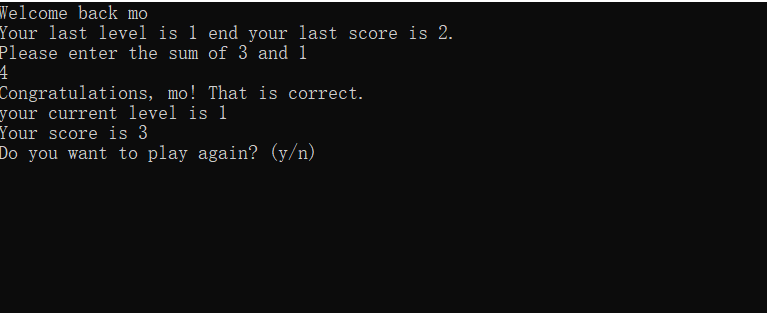
（2）进入菜单



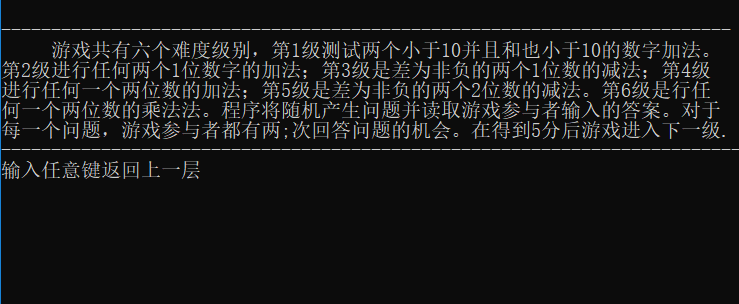
（3）输入1开始游戏



（4）输入2继续上一次游戏

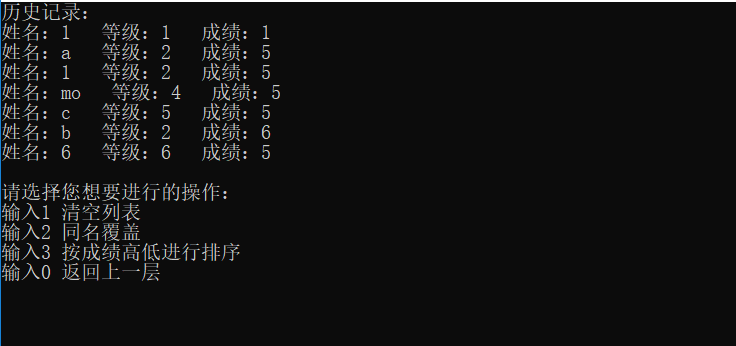


（5）输入3可显示游戏说明

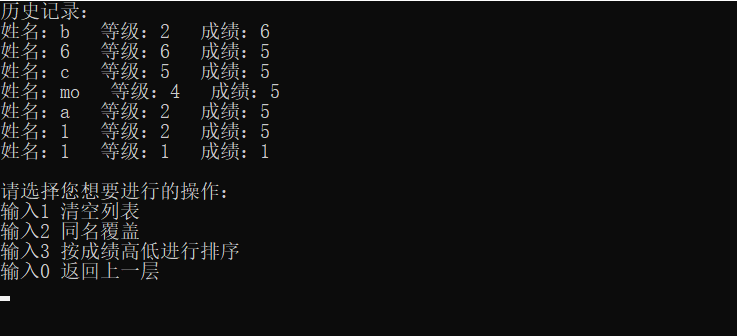


（5）输入4可查看历史记录

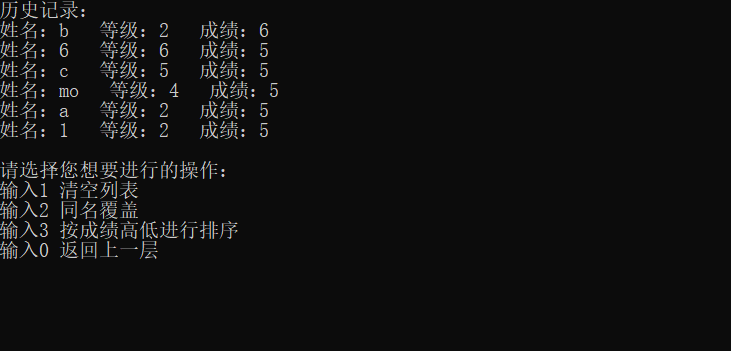
历史记录



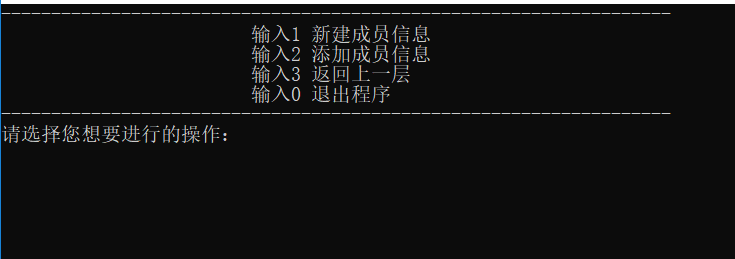
输入3排名



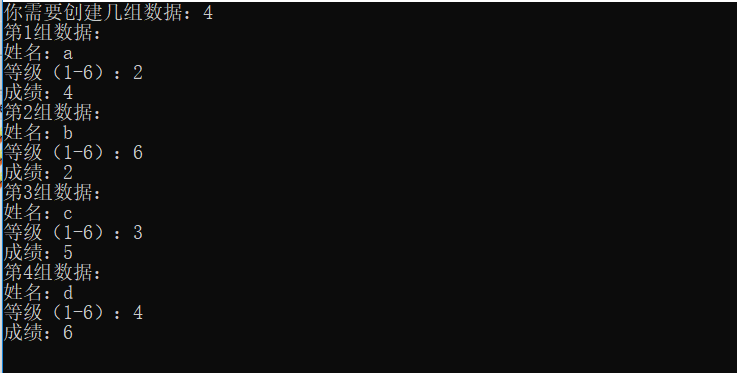
输入2同名覆盖

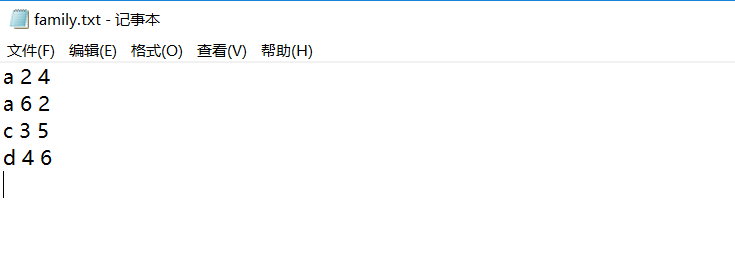


（6）输入5创建新的记录

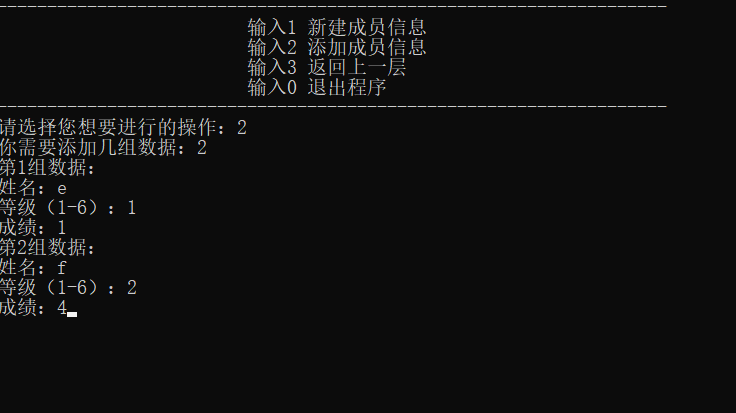


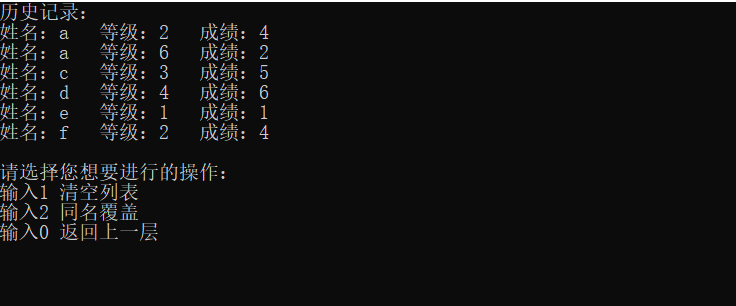
新建成员





添加成员：

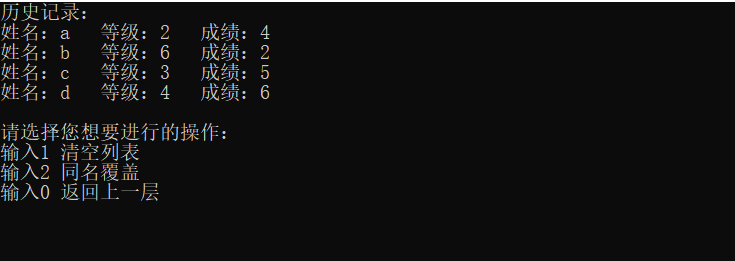




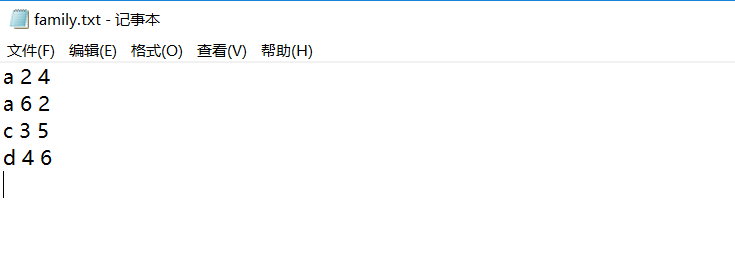


（7）输入6可修改成绩

上次创建的记录

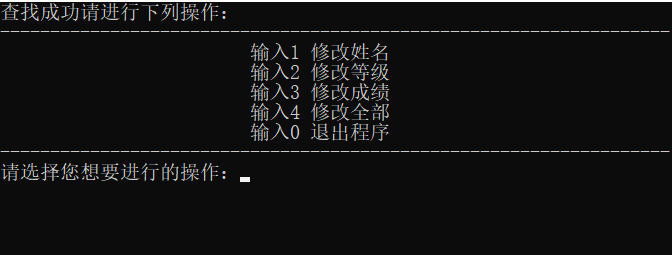


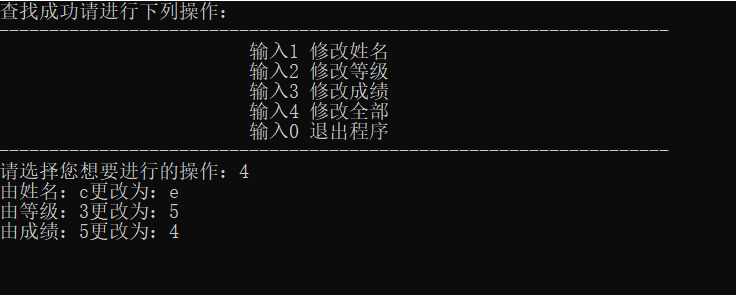
在文件中

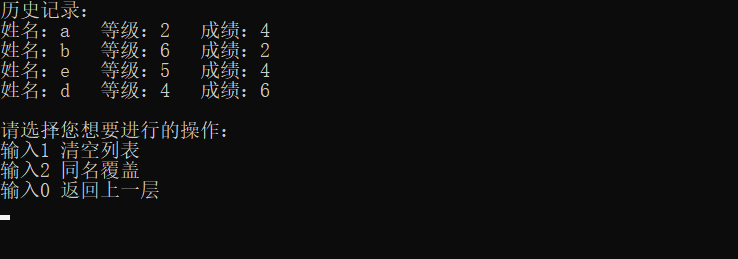


修改c

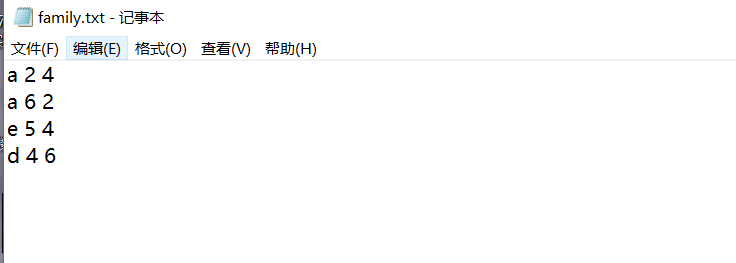




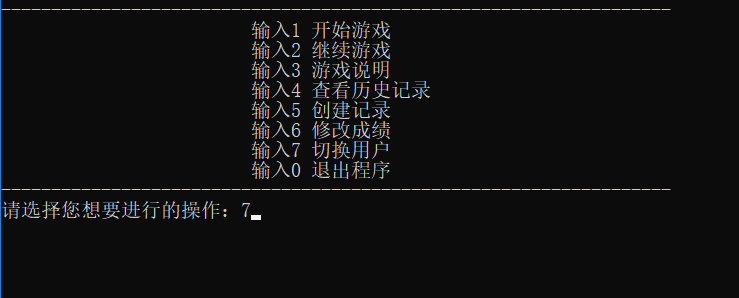


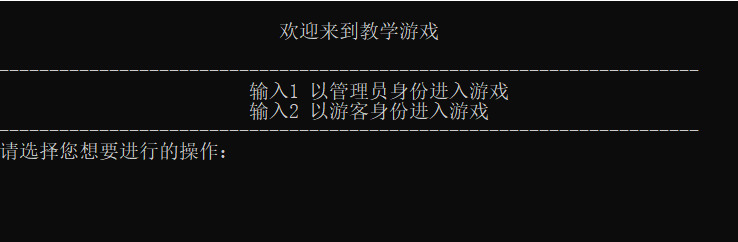


在family文件中

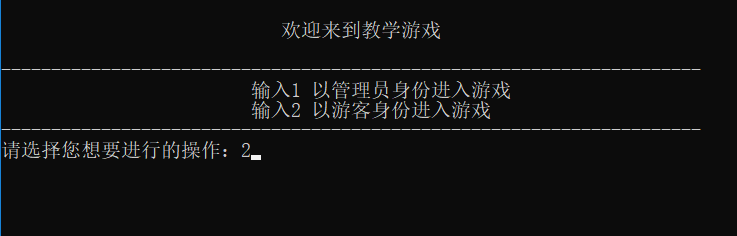


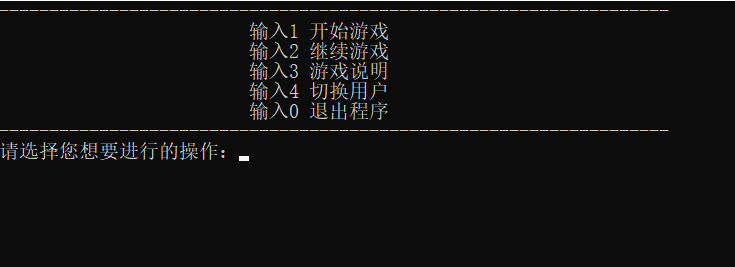
（8）输入7可切换用户





**3.以游客身份进入游戏**





# 六、心得体会

**（1）实验结果分析：**

1、教学游戏基础功能基本实现，程序运行成功。

2、程序基本满足实验的要求。

3、扩展了不少的内容使得教学游戏更加充实切合实际。

**（2）总结：**

1、有创新点，但功能实现方面略还有些不足。

2、将数据结构中的链表的操作和文件操作相结合，灵活运用了所学内容。

3、代码组织结构不够合理，部分代码过于冗余。

4、用户体验做得不够好，程序还未实现对用户的其他错误操作进行处理，比如菜单选项用户错误输入英文字母的情况未做处理。

5、踏踏实实做好每一件事。一项实验是由无数细小的环节组成的，每一个细节处理的好坏都可能影响到最后实验的成败。总之，事无巨细，一切从小事做起，从点滴做起。

**（3）对面向对象程序设计的理解**

1、主要是运用类和对象进行封装。

2、个人感觉文件的操作对于面对对象程序设计很重要。

3、在设计时，可重用现有的，在以前的项目的领域中已被测试过的类使系统满足业务需求并具有较高的质量。

4、在软件开发时，根据设计的需要对现实世界的事物进行抽象，产生类。

# 七、参考文文献

【1】《数据结构-用C语言描述》.谭策善、李龙澎.出版社：高等教育出版社.2019.12.

【2】《C++面对对象程序设计（第二版）》.谭浩南.出版社：清华大学出版社.2018.12.