

**信息与软件工程学院**

**项目报告**

课程名称： 程序设计项目实践（PBLF）

学 期： 2021-2022第1学期

项目名称： 烧脑游戏串烧

授课教师： 陈伟

|  |  |  |
| --- | --- | --- |
| 序号 | 学号 | 姓名 |
| 1（组长） | 2021090906016 | 刘芮利 |
| 2 | 2021090906020 | 张庭恺 |
| 3 | 2021090906002 | 赵晗 |
| 4 | 2021090906010 | 钟城伟 |
| 5 | 2021090906008 | 黄培坤 |

目录

[1 项目简介 3](#_Toc91795855)

[1.1 考核方式 3](#_Toc91795856)

[1.2 项目题目及内容简介 3](#_Toc91795857)

[1.3 项目组成员与分工 4](#_Toc91795858)

[2 需求分析 4](#_Toc91795859)

[2.1 选题的依据 4](#_Toc91795860)

[2.2 功能需求 5](#_Toc91795861)

[3 系统设计 6](#_Toc91795862)

[3.1 总体设计 6](#_Toc91795863)

[3.2 模块设计 7](#_Toc91795864)

[4 系统实现 8](#_Toc91795865)

[4.1 主函数 8](#_Toc91795866)

[4.2 其他函数 1](#_Toc91795867)4

[5 测试 1](#_Toc91795868)6

[6 总结 1](#_Toc91795869)7

# 1 项目简介

## 1.1 考核方式

总成绩 = 项目和项目文档成绩(40%) + 汇报幻灯片成绩(20%)

+ 表达能力(20%) + 团队合作(20%)

## 1.2 项目题目及内容简介

* 项目题目：烧脑游戏串烧（小游戏合集）
* 内容简介：项目成果是一款益智游戏主题的小游戏合集，具体包括猜数游戏、猜拳游戏、取石子博弈、吃豆人、长城游戏、俄罗斯方块，每一个游戏有多个关卡和不同模式可供选择。支持多用户注册、登录和PK，天梯排名系统可以显示所有用户的游戏得分，同时每个游戏也可以查看历史得分记录排名。游戏数据使用文件保存和读取，为了保障数据安全，程序数据采用RSA非对称加密算法对游戏数据进行加密。每个游戏的简要介绍如下：
  + - 猜数游戏：玩家猜测程序随机生成的数字，有一定次数限制。根据猜测次数和用时计算得分。
    - 猜拳游戏：玩家与电脑进行猜拳，通过模拟石头剪刀布来实 现。玩家可进行多次游戏来对自己今天的运气进行没有科学依据的判断。
    - 取石子博弈：玩家和AI每次可从任意一堆石子里取出任意多枚 石子扔掉，可以取完，不能不取，每次只能从一堆里取。取走最 后一颗石子的一方获得胜利。根据AI难度、撤回次数和用时计算得分。
    - 吃豆人：玩家可使用键盘操控吃豆人躲避敌人攻击获取分数，可通过增幅道具增加速度
    - 长城游戏：在NN方阵中有N个‘/’，请玩家数出移动‘/’使之形成一条直线“长城”的最小步数。答案正确则获胜。有三次回答机会，并计时，所用时间越少或回答关卡数越靠后分数越高。
    - 俄罗斯方块：玩家可以进行四种难度模式选择，难度越大即俄罗斯方块下落速度越快，满行即清除并且记1000分

## 1.3 项目组成员与分工

成员：刘芮利、赵晗、黄培坤、张庭恺、钟城伟

刘芮利：游戏框架设计、头文件设计与编写、代码后期合成、登录/注册系统、关卡切换、天梯排名系统、加解密模块、数据存储、猜数游戏编写、取石子博弈游戏编写

赵晗：猜拳游戏编写（使用Windows.h头文件实现对system和sleep函数的应用，使程序更具有现实性和灵动性;独立设计case\_win和echoinput两个函数对主函数进行“瘦身”;利用时间作种子生成随机数实现真随机）

黄培坤：编写小游戏俄罗斯方块，学习并使用颜色函数，解决碰撞判断法性问题以及研究方块旋转逻辑

张庭恺：吃豆人游戏的编写

钟城伟：长城游戏编写（包括1.最短步数的计算，等价于最佳匹配，使用了KM算法，大大缩短了时间复杂度，由缩减为。2.作了获胜或失败后的美化。3.追踪生成答案方阵。4.利用字符数组读入用户输入，再进行判断，增强程序的排错能力，避免出现bug。5.使用时间函数，以达到计时目的。6.巧妙使用多个goto语句，以达成选择性循环。）

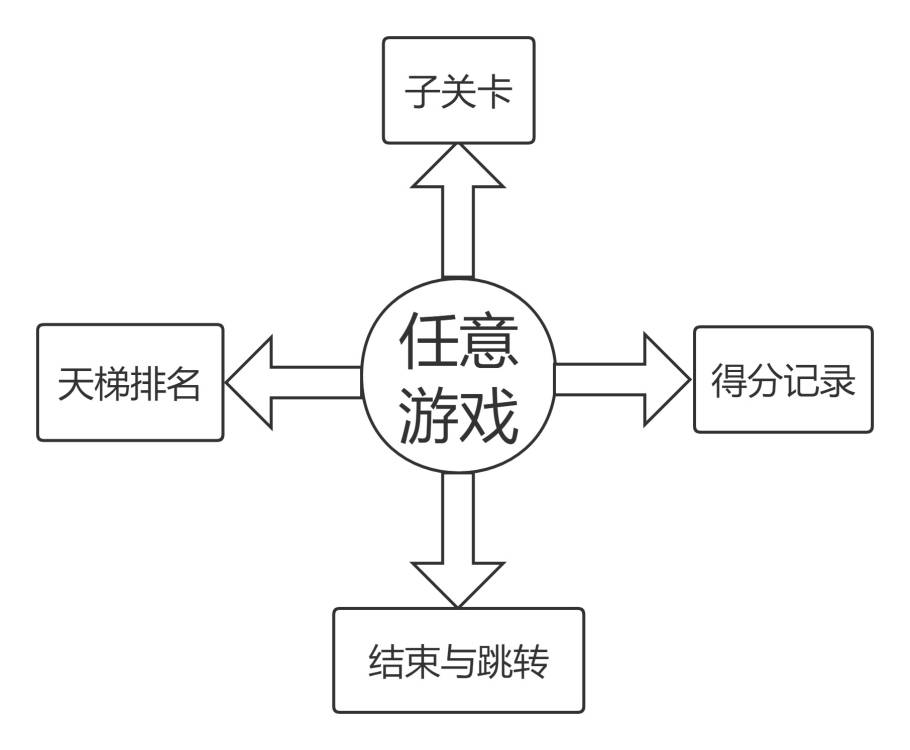
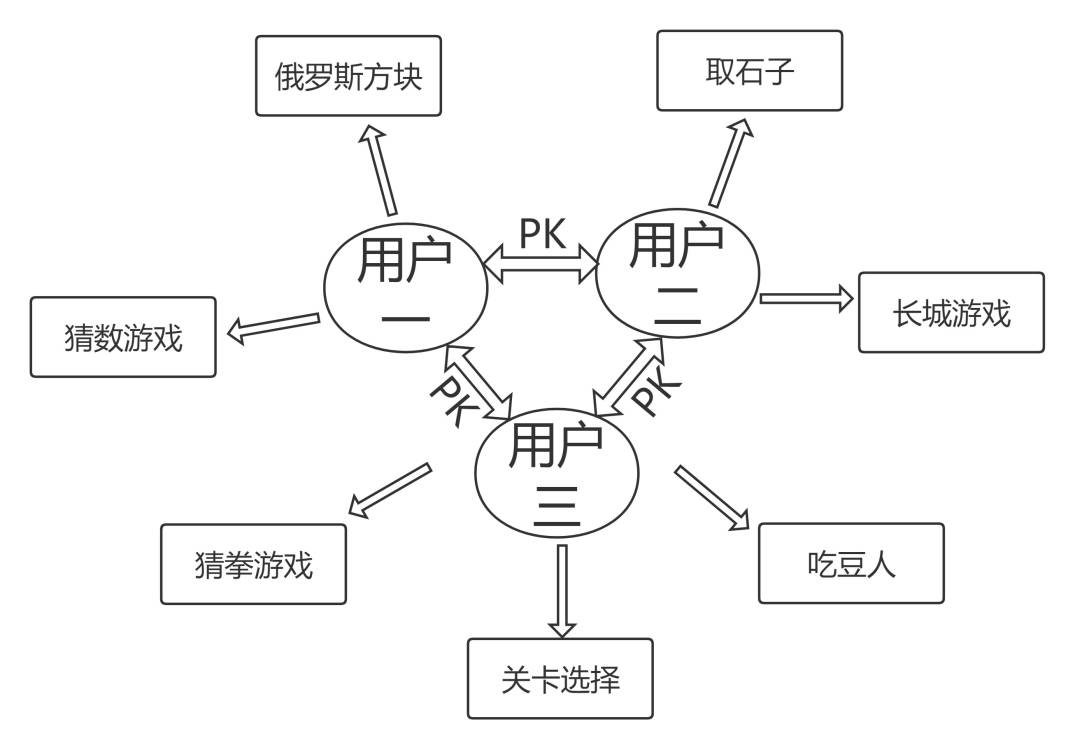
# 2 需求分析

## 2.1 选题的依据

* 俄罗斯方块、Nim博弈、吃豆人等都是经典小游戏，通过这学期所学知识，复现经典小游戏提高C语言代码能力，同时可供闲暇之余消遣娱乐
* 多用户登录/注册、用户得分排名、历史得分记录等是小游戏体验的基本需要，项目对这些功能进行了简易还原，以此提高了游戏的可玩性和竞技性
* 为防止游戏数据被恶意篡改，数据存储安全对程序正常、可靠的运行至关重要。游戏采用现今最广泛运用的RSA非对称加密算法对数据进行加解密，保障数据安全

### 2.2 功能需求

下面通过UML示例图说明程序功能：



# 3 系统设计

## 3.1 总体设计

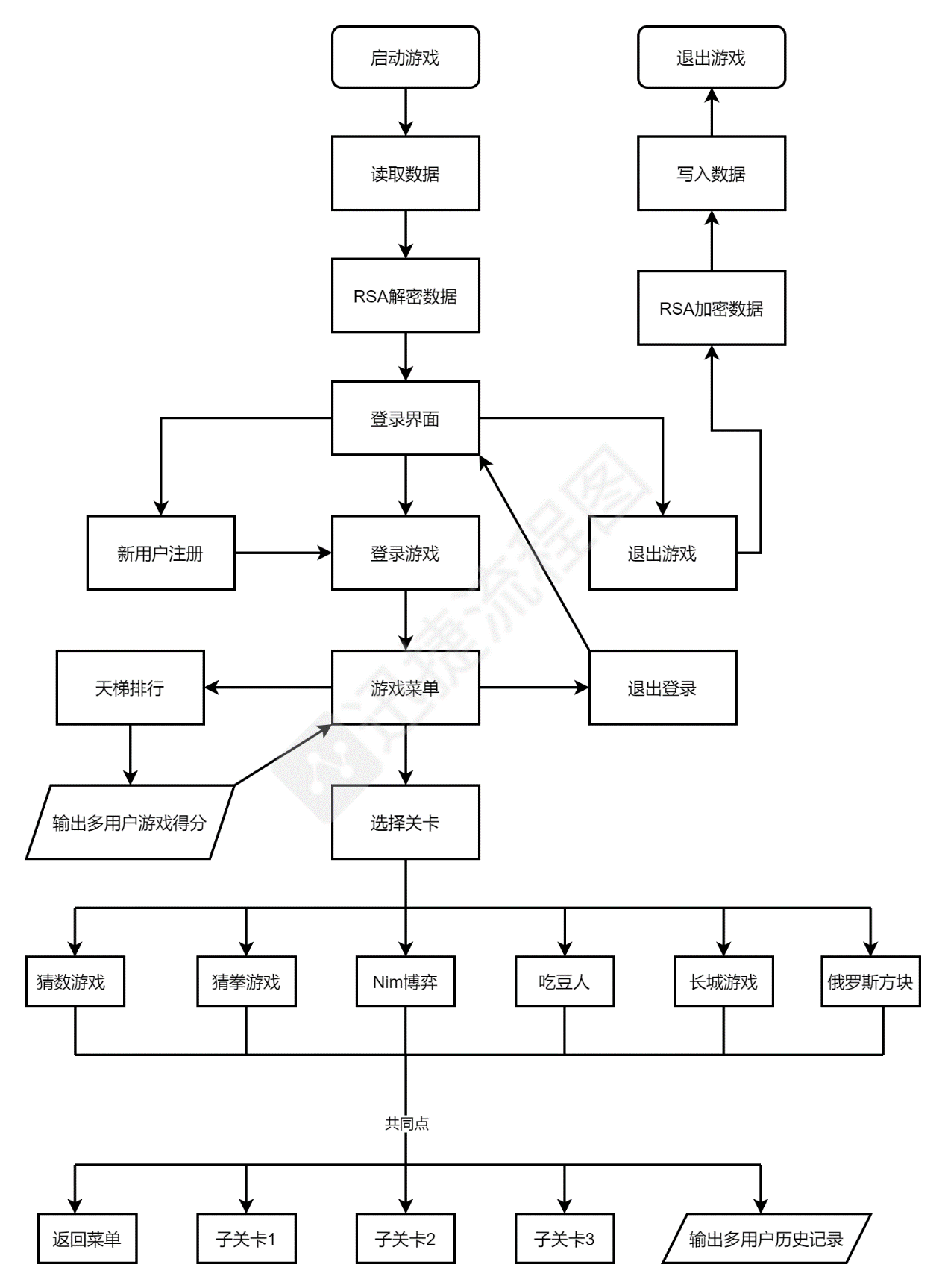
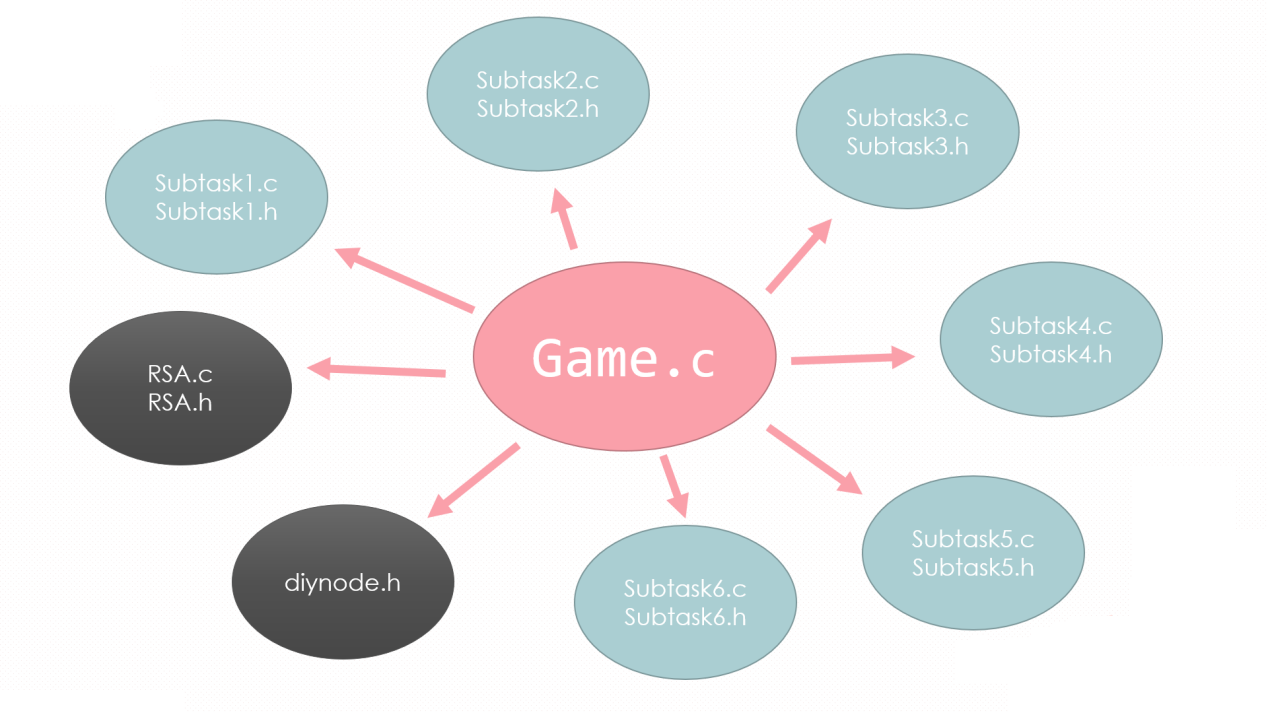


图3-1

## 3.2 模块设计



* **Subtask.h**

1. int Max(int, int); *//求最小值函数*
2. int Min(int, int); *//求最大值函数*
3. void Switchtask(); *//切换关卡函数*
4. void Subtask3(); *//你的游戏函数入口，这里以Subtask3举例*
5. void ShowRank(int); *//展示对应游戏用户排名*

* **Subtask.c**

1. #include "Subtask2.h" *//引入你的头文件，这里以Subtask2举例*
2. #include "diynode.h"  *//引入自定义结构体类型,*
3. #include<stdio.h>
4. extern node Data[]; *//共享游戏数据Data*
5. extern int id, taskid; *//共享id, taskid, id为当前用户对应的数字序号, 用作Data[]下标, taskid表示当前正在进行或即将进入的关卡, 你很可能会用到*
6. *//你可以在下面写函数，定义变量等等*
7. void Subtask2(){ *//下面将是你的游戏入口, 确保return后将taskid更改为合适的值, 1~Gamecnt表示对应关卡, -1表示退出游戏*
8. puts("Subtask2:"); *//这只是一个演示测试而已, 建议删除*
9. Data[id].taskid = Data[id].taskid > 2 ? Data[id].taskid : 2; *//更新用户的最高关卡记录*
10. Switchtask(); *//切换关卡, 可在退出时调用*
11. }

* **Diynode.h**

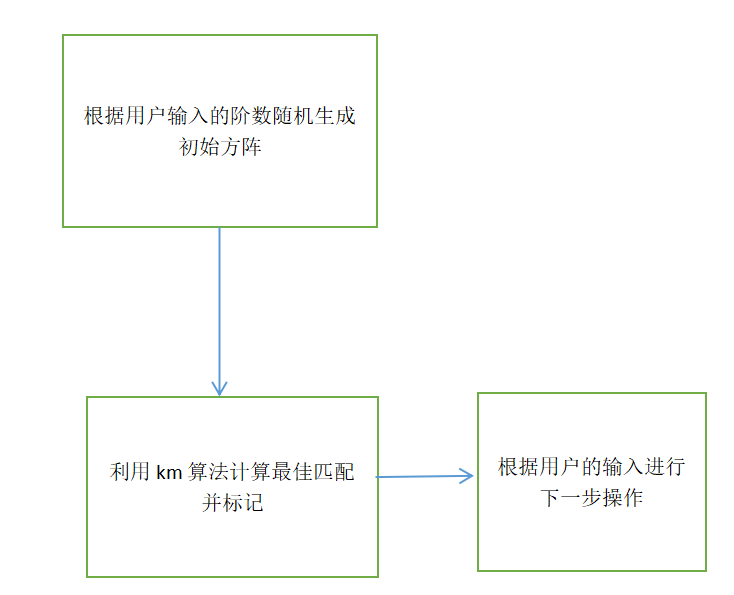
1. #define GameCnt 8
2. typedef struct Node{ *//存储游戏数据的自定义结构体*
3. char username[12];
4. char password[35]; int taskid;
5. int data[GameCnt+5][10];
6. }node;

**注：以上模块每个成员共用的代码模板，便于统一共享游戏数据和后期代码整合**

# 4 系统实现

## 4.1 主函数

用程序流程图说明实现思路



关键代码分析

问题描述；移动任意摆放的n个/，使其在n\*n的方阵中形成一条可以抵御敌人的长城。请求出所需移动/的步数之和的最小值。

# 问题分析：首先，我们发现/在移动过程中是否重合对结果没有影响。这是因为我们并不关心哪一个/移动到了对应的位置，这样，当移动A需要经过B所在的位置时，我们可以让B前去A要去的位置，而让A留在B所在的位置。于是，我们不用考虑两个/移动到同一个位置上的错误情况。于是，我们的程序变为解决对于2n+2个最终情况的最佳匹配求解。为此，我们将所有距离取相反数，作为权，则将问题转化为二分图的最佳完美匹配问题。

核心算法：KM算法

核心算法原理：将初始位置的集合X与最终位置的集合Y视为一个二分图。然后，每个顶点设置一个顶标，使得X、Y的顶标之和大于等于对应顶点的权。然后，我们寻找对于X中每一个顶点的相等子图中的增广路。找到增广路即为找到了X中对应点的一个权等于顶标之和的边。如果对于X中一点没有增广路，则变换顶标，增加相等子图中的边数。最后，当每一个X中的点都找到增广路时，即找到了最佳匹配。

其余请看具体注释

|  |
| --- |
| 1. int main() 2. { 3. printf( "╭===================○●○●==================╮\n\ 4. │  ★ ☆欢迎来到Subtask5: 长城游戏☆ ★       │\n\ 5. ╰=======○●○●==============================╯\n游戏规则:移动任意摆放的n个/，使其在n\*n的方阵中形成一条可以抵御敌人的直线长城。请求出所需移动/的步数之和的最小值。\n" ); 6. int sas=0,sa=0,ssa,score[3]={0},xuan=0; 7. float ze,guanka; 8. char str[100]={'\0'},strr[100]={'\0'},strrr[100]={'\0'}; 9. done: 10. fflush(stdin); 11. int cishu=2; 12. printf( "1.选择关卡\n2.查看历史分数\n3.退出游戏\n" ); 13. for(sa=0;;){scanf("%c",&str[sa]); 14. if(str[sa]=='\n') break; 15. sa++;*//利用字符数组进行读入，方便判定非法输入* 16. } 17. ze=(str[0]-'0'); 18. if (str[1]!='\n'){ 19. printf("非法输入，请重输。\n");goto done; 20. } 21. if (ze==1){ 22. tryagain: 23. printf("总共三关，你已打通%d关，所以你可以选择玩前%d关。\n请输入你的选择（1、2、3）\n",xuan,xuan==3?3:xuan+1); 24. for(sas=0;;){scanf("%c",&strr[sas]); 25. if(strr[sas]=='\n') break; 26. sas++; 27. } 28. guanka=(strr[0]-'0'); 29. if (strr[1]!='\n'){ 30. printf("非法输入，请重新输入。\n");goto tryagain; 31. } 32. if(guanka>(xuan==3?3:xuan+1)||guanka<=0){ 33. printf("输入非法，请重新输入。\n");goto tryagain; 34. } 35. N=guanka\*2+1;} 36. else if (ze==2){printf("总分\t第一关分数\t第二关分数\t第三关分数\n%d \t%d       \t%d       \t%d\n",score[0]+score[1]+score[2],score[0],score[1],score[2]); 37. goto done;} 38. else if (ze==3){return;} 39. else { 40. printf("非法输入，请重输。\n");goto done;} 42. n = N; 43. int X, Y, x, y, i, j, k, s,ss, wei; 44. bool pan; 45. char r[N][N], xian[N][N]; 46. for ( X = 0; X < N; X++ ) 47. for ( Y = 0; Y < N; Y++ ) 48. { 49. xian[X][Y] = '\*'; 50. r[X][Y]  = '\*'; 51. }*//初始化数组* 52. int A, B; 53. char m; 54. int h=0; 55. int P[N + 1][2]; 56. srand( (unsigned) time( NULL ) ); 57. for ( i = 1; i <= N; i++ ) 58. { 59. A = rand() % N; 60. B = rand() % N; 61. while ( r[A][B] != '\*' ) 62. { 63. A = rand() % N; 64. B = rand() % N; 65. } 66. r[A][B] = '/'; 67. P[i][0] = A + 1; P[i][1] = B + 1; 68. }*//随机生成n个'/'* 69. for ( x = 0; x < N; x++ )*//显示* 70. { 71. for ( y = 0; y < N; y++ ) 72. { 73. printf( "%c ", r[x][y] ); 74. if ( y == N - 1 ) 75. printf( "\n" ); 76. } 77. } 78. ans = -(n \* n + n); 79. for ( i = 1; i <= n; i++ )*//利用4个for循环对四种情况进行计算与比较* 80. { 81. for ( j = 1; j <= n; j++ ) 82. for ( k = 1; k <= n; k++ ) 83. a[j - 1][k - 1] = -dis( P[j][0] - i, P[j][1] - k ); 84. pan = work(); 85. if ( pan == true ) 86. { 87. wei = i; 88. } 89. } 90. for ( i = 1; i <= n; i++ ) 91. { 92. for ( j = 1; j <= n; j++ ) 93. for ( k = 1; k <= n; k++ ) 94. a[j - 1][k - 1] = -dis( P[j][0] - k, P[j][1] - i ); 95. pan = work(); 96. if ( pan == true ) 97. { 98. wei = -i; 99. } 100. } 101. for ( j = 1; j <= n; j++ ) 102. for ( k = 1; k <= n; k++ ) 103. a[j - 1][k - 1] = -dis( P[j][0] - k, P[j][1] - k ); 104. pan = work(); 105. if ( pan == true ) 106. { 107. wei = 0; 108. } 109. for ( j = 1; j <= n; j++ ) 110. for ( k = 1; k <= n; k++ ) 111. a[j - 1][k - 1] = -dis( P[j][0] - k, P[j][1] - (n - k + 1) ); 112. pan = work(); 113. if ( pan == true ) 114. { 115. wei = 101; 116. } 117. if ( wei >= 1 && wei <= n ) 118. { 119. for ( i = 0; i < N; i++ ) 120. xian[wei - 1][i] = '/'; 121. } 122. if ( (-wei) >= 1 && (-wei) <= n ) 123. { 124. for ( i = 0; i < N; i++ ) 125. xian[i][-wei - 1] = '/'; 126. } 127. if ( wei == 0 ) 128. { 129. for ( i = 0; i < N; i++ ) 130. xian[i][i] = '/'; 131. } 132. if ( wei == 101 ) 133. { 134. for ( i = 0; i < N; i++ ) 135. xian[i][N - i - 1] = '/'; 136. }*//生成答案方阵* 137. start =time(NULL); 138. retry: 139. printf( "输入你的答案" ); 140. for(ssa=0;;){scanf("%c",&strrr[ssa]); 141. if(strrr[ssa]=='\n') break; 142. ssa++; 143. } 144. s=(strrr[0]-'0');ss=(s\*10)+(strrr[1]-'0'); 145. if ( strrr[1]<='9'&&strrr[1]>='0' ) goto turn; 146. if (strrr[1]!='\n'){ 147. printf("非法输入，请重输。\n");goto retry; 148. } 149. turn: 150. if ( s == -ans||ss==-ans ) 151. { printf( "                              \_       \_ \n\ 152. (\_)     | |\n\ 153. \_   \_  \_\_\_  \_   \_  \_\_      \_\_\_ \_ \_\_ | |\n\ 154. | | | |/ \_ \\| | | | \\ \\ /\\ / / | '\_ \\| |\n\ 155. | |\_| | (\_) | |\_| |  \\ V  V /| | | | |\_|\n\ 156. \\_\_, |\\\_\_\_/ \\\_\_,\_|   \\\_/\\\_/ |\_|\_| |\_(\_)\n\ 157. \_\_/ |                                 \n\ 158. |\_\_\_/   \n" );   end =time(NULL); 159. printf("用时%d秒\n",(int)difftime(end,start)) ; 160. if(score[((int)guanka)-1]<((n+cishu)\*10+1000/(int)difftime(end,start))) 161. score[((int)guanka)-1]=((n+cishu)\*10+1000/(int)difftime(end,start)); 162. if(xuan<(int)guanka) xuan=(int)guanka; 163. printf("本局得分：%d\n",((n+cishu)\*10+1000/(int)difftime(end,start))) ; 164. } 165. else{ 166. printf( "                     \_ \n\ 167. | |                 \n\ 168. \_   \_  \_\_\_  \_   \_  | | \_\_\_  \_\_\_  \_\_\_   \n\ 169. | | | |/ \_ \\| | | | | |/ \_ \\/ \_\_|/ \_ \\  \n\ 170. | |\_| | (\_) | |\_| | | | (\_) \\\_\_ \\  \_\_/\_ \n\ 171. \\\_\_, |\\\_\_\_/ \\\_\_,\_| |\_|\\\_\_\_/|\_\_\_/\\\_\_\_(\_)\n\ 172. \_\_/ |  \n\ 173. |\_\_\_/ \n" ); 174. while(cishu>0){printf("总共3次答题机会，还有%d次机会\n",cishu);cishu--;goto retry;} 175. printf("你已用完所有的答题机会，是否查看答案(y/n)\n"); 176. scanf( "\n%c", &m ); 177. } 178. if ( m == 'y' ) 179. { 180. printf( "正确答案是%d\n", -ans ); 181. for ( x = 0; x < N; x++ ) 182. { 183. for ( y = 0; y < N; y++ ) 184. { 185. printf( "%c ", xian[x][y] ); 186. if ( y == N - 1 ) 187. printf( "\n" ); 188. }*//显示答案方阵* 189. } 190. } 191. tail: 192. fflush(stdin); 193. printf( "（1.x）继续游戏\n" ); 194. printf( "（2.x）显示分数\n" ); 195. printf( "（大于等于3或字母或负数） 退出游戏\n" ); 196. printf("请输入你的选择\n"); 197. scanf( "\n%d", &h ); 198. if ( h == 1 ) 199. goto done; 200. else if(h==2) 201. { 202. printf("总分\t第一关分数\t第二关分数\t第三关分数\n%d \t%d       \t%d       \t%d\n",score[0]+score[1]+score[2],score[0],score[1],score[2]); 203. goto tail;} 204. else 205. return; 207. } |

## 4.2 其他函数

1. #include <stdio.h>
2. #include <stdlib.h>
3. #include <time.h>
4. #include <stdbool.h>
5. #include <string.h>
6. #define MaxN 100
7. time\_t start,end;
8. int N, n, ans;
9. int a[MaxN + 1][MaxN + 1], Lx[MaxN + 1], Ly[MaxN + 1], matchY[MaxN + 1];
10. bool usedX[MaxN + 1], usedY[MaxN + 1];
11. bool path( int r )*//判断是否有增广路*
12. {
13. int i;
14. usedX[r] = true;
15. for ( i = 0; i < n; ++i )
16. if ( Lx[r] + Ly[i] == a[r][i] )
17. if ( !usedY[i] )
18. {
19. usedY[i] = true;
20. if ( matchY[i] < 0 || path( matchY[i] ) )
21. {
22. matchY[i] = r; return(true);
23. }
24. }
25. return(false);
26. }
27. bool work()*//利用KM算法求最佳匹配，并标记*
28. {
29. int i, j, k, res = 0;
30. for ( i = 0; i < n; ++i )
31. {
32. Lx[i] = -2147483647;
33. for ( j = 0; j < n; ++j )
34. if ( a[i][j] > Lx[i] )
35. Lx[i] = a[i][j];
36. }
37. memset( Ly, 0, sizeof(Ly) );*//初始化顶标Lx,Ly，使其满足顶标之和大于等于边权*
38. memset( matchY, -1, sizeof(matchY) );
39. for ( i = 0; i < n; ++i )
40. for (;; )*//寻找增广路*
41. {
42. memset( usedX, false, sizeof(usedX) );
43. memset( usedY, false, sizeof(usedY) );
44. if ( path( i ) )
45. break;
46. int delta = 2147483647, v;
47. for ( j = 0; j < n; ++j )
48. if ( usedX[j] )
49. for ( k = 0; k < n; ++k )
50. if ( !usedY[k] )
51. {
52. v = Lx[j] + Ly[k] - a[j][k];
53. if ( v < delta )
54. delta = v;
55. }
56. for ( j = 0; j < n; ++j )
57. if ( usedX[j] )
58. Lx[j] -= delta;
59. for ( j = 0; j < n; ++j )
60. if ( usedY[j] )
61. Ly[j] += delta;*//变换，增加相等子图中的边数*
62. }
63. for ( i = 0; i < n; ++i )
64. if ( matchY[i] >= 0 )
65. res += a[matchY[i]][i]; *//求和*
66. if ( res > ans )
67. {
68. ans = res; return(true);*//比较，标记最佳解*
69. }else return(false);
70. }
71. int dis( int x, int y )*//求两点间的距离*
72. {
73. if ( x \* y >= 0 )
74. x += y;
75. else x -= y;
76. if ( x > 0 )
77. return(x);
78. else return(-x);
79. }

# 5 测试

程序运行理想，无bug。

# 6 总结

**钟城伟-2021090906010：**

1.学到了什么

我学到了如何合成多个源代码，KM算法，随机生成点的方法，计算距离的方法，判定输入内容的方法，以及如何分析问题的方法。这次实践活动让我认识到了查阅资料与团队合作的重要性。

2.痛点和难点

计算最佳匹配一开始不知道怎么弄好，因为如果暴力计算的话时间复杂度为n\*n!,很慢，甚至不知道最佳匹配这个名词，在经过长时间上网查找资料后才得出方案。

还有为了防止非法输入，修复了N个bug。

3.自己的贡献

编写了程序的一个游戏，代码300行左右。

4.如何与他人合作

写好一个个初稿后与同学们进行讨论，对游戏的功能进行一步步优化。