**C语言项目报告**

**项目名称：改进版三子棋小游戏\_\_\_\_\_\_\_\_\_\_\_**

**项目成员：\_\_\_\_姜泓源\_\_\_\_\_\_\_**

**填写日期：\_\_2020.9.9\_\_\_\_\_\_\_\_\_**

**摘要（Abstract）**

在传统三子棋上进行改进，避免因为棋盘填满而平局的问题，每位玩家仅有三枚棋子，在放下第三枚棋子后，依次拿起第一枚和第二枚继续游戏，直到一方完成三连结束。实现自创棋类的人机对战功能，需要完全自创算法，实现全新的游戏规则。需要的功能包括两步以上的推演能力，对三枚棋子进行正确的删减和循环。

**问题描述（Problem Statement）**

作为一个小游戏，需要有做到操作简便、游戏本身复杂度高、人机有足够的难度。操作拟定使用方向键移动光标，空格进行落子。游戏设计上具有较高的复杂度，玩家需要记忆棋子的放置顺序，并且需要对两步及以后的局势有一定的判断能力。对电脑难度的要求：能正确的处理一步之内的棋局（完成三连或防止对方三连），对于两步以上的棋局有一定的判断能力。

**组内分工（Group Division）**

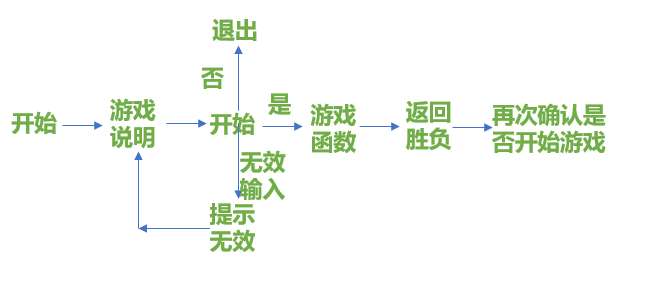
姜泓源 约1100行代码 单人完成

**分析（Analysis）**

总体框架为选择开始游戏，进行游戏，判断胜负的循环。开始游戏前需要对游戏界面进行设定，显示游戏说明。游戏函数包括电脑移动、删除电脑移动之前位置的棋子，判断胜负，玩家移动，需要键盘移动，删除棋子，判断胜负。其中键盘移动和删除棋子需要记录棋子的原有位置，在程序中使用数组完成存储功能。判断胜负的模块根据判断胜负的部分返回的信息，会判断游戏继续或显示胜负情况。以上完成一个循环。

**设计（Design）**

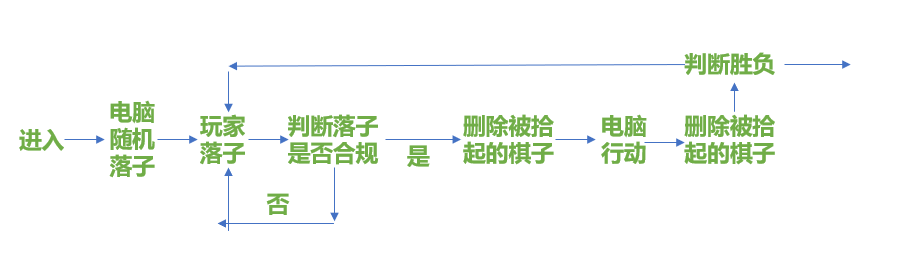
主函数：



游戏函数：

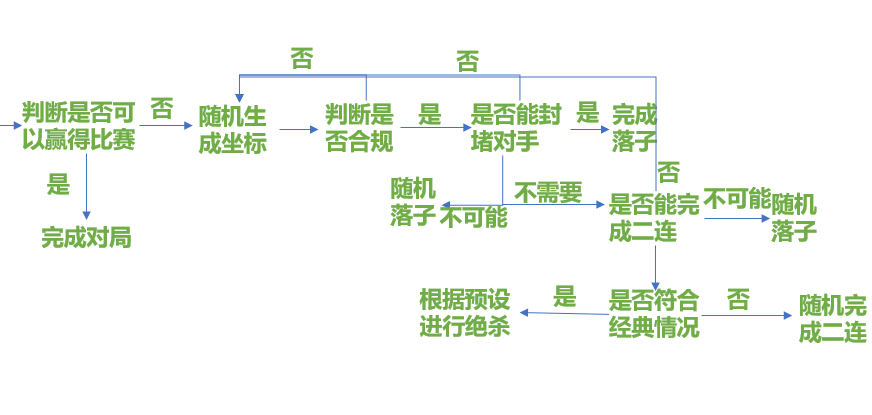
删除棋子部分使用数组记录棋子顺序，数组定义为一个较大数值，抱着不会被超出。

玩家落子的光标选择使用覆盖式，可以避免闪烁，定义一个临时二维数组来记录被覆盖掉的棋子信息。



人机逻辑函数：

人机逻辑是对随机生成坐标进行筛选，这样就有足够的随机性以及便利的难度调节。可以通过调节循环次数的最大值来改变难度。



**实施（Implementation）**

主函数：

int main()

{

modeset(33, 25);

int c = 0;

printf("改进版三子棋小游戏\n\n游戏说明：\n双方拥有三枚棋子，下完三枚棋子后，\n按顺序依次拾起最初下的棋子继\n续游戏，当一方完成三连时获胜。\n\n");

printf("操作方法:\n方向键移动棋子，“空格”键确定\n落子\n\n\n");

do {

//颜色

system("color E0");

SetConsoleTitleA("改进版三子棋");

menu();

scanf\_s("%d", &c);

switch (c) //用switch语句来实现进入游戏或者退出游戏

{

case 1:

game();

break;

case 2:

exit(0);

default:

printf("Select error!!!\n");

}

} while (1);

system("pause");

return 0;

}

**游戏函数：**

void game()

{

c = 1;

d = 1;

char board[ROW][COL]; char board2[ROW][COL];

memset(board, ' ', ROW \* COL); //将数组全部初始化为空格

char ret;

do {

displayBoard(board, COL); //继续显示棋盘

compMove(board, COL);

del2(board, COL);

displayBoard(board, COL); //继续显示棋盘

ret = isWin(board, COL);

if (ret != ' ')

{

break;

}

p = 1;

q = 1;

int abc[5][5];

int j, k;

abc[p][q] = board[1][1];

j = 0;

while (j==0)

{

board[p][q] = '+';

displayBoard(board, COL); //继续显示棋盘

board[p][q] =abc[p][q] ;

char ch = getch();

switch (ch)

{

abc[p][q] = board[p][q];

case 27://esc

exit(0);

case 72://上

p--;

if (p < 0)p = 2;

abc[p][q] = board[p][q];

break;

case 80://xia

p++;

if (p >2)p = 0;

abc[p][q] = board[p][q];

break;

case 75://zuo

q--;

if (q < 0)q = 2;

abc[p][q] = board[p][q];

break;

case 77://you

q++;

if (q > 2)q = 0;

abc[p][q] = board[p][q];

break;

case 32://空格

if (board[p][q] == ' ')

{

board[p][q] = 'o';

a[c] = p;

b[c] = q;

j = 1;

}

else {

j = 0;

}

break;

}

}

del1(board, COL);

ret = isWin(board, COL); //是否产生结果

if (ret != ' ')

{

break;

}

displayBoard(board, COL);

d = d + 1;

c = c + 1;

} while (ret == ' ');

if (ret == 'o') //返回值为o则表示玩家赢了

{

printf("nice,you win\n");

}

else if (ret == 'x') //返回值为x则表示电脑赢了

{

printf("you lose,try again\n");

}

}

**测试（Test）**

