//随机生成食物

void RandFood()

{

int i, j;

do

{

//随机生成食物的横纵坐标

i = rand() % ROW;

j = rand() % COL;

} while (face[i][j] != KONG); //确保生成食物的位置为空，若不为空则重新生成

face[i][j] = FOOD; //将食物位置进行标记

color(12); //颜色设置为红色

CursorJump(2 \* j, i); //光标跳转到生成的随机位置处

printf("●"); //打印食物

}

//判断得分与结束

void JudgeFunc(int x, int y)

{

//若蛇头即将到达的位置是食物，则得分

if (face[snake.y + y][snake.x + x] == FOOD)

{

snake.len++; //蛇身加长

grade += 10; //更新当前得分

color(7); //颜色设置为白色

CursorJump(0, ROW);

printf("当前得分:%d", grade); //重新打印当前得分

RandFood(); //重新随机生成食物

}

//若蛇头即将到达的位置是墙或者蛇身，则游戏结束

else if (face[snake.y + y][snake.x + x] == WALL || face[snake.y + y][snake.x + x] == BODY)

{

Sleep(1000); //留给玩家反应时间

system("cls"); //清空屏幕

color(7); //颜色设置为白色

CursorJump(2 \* (COL / 3), ROW / 2 - 3);

if (grade > max)

{

printf("恭喜你打破最高记录，最高记录更新为%d", grade);

WriteGrade();

}

else if (grade == max)

{

printf("与最高记录持平，加油再创佳绩", grade);

}

else

{

printf("请继续加油，当前与最高记录相差%d", max - grade);

}

CursorJump(2 \* (COL / 3), ROW / 2);

printf("GAME OVER");

while (1) //询问玩家是否再来一局

{

char ch;

CursorJump(2 \* (COL / 3), ROW / 2 + 3);

printf("再来一局?(y/n):");

scanf("%c", &ch);

if (ch == 'y' || ch == 'Y')

{

system("cls");

main();

}

else if (ch == 'n' || ch == 'N')

{

CursorJump(2 \* (COL / 3), ROW / 2 + 5);

exit(0);

}

else

{

CursorJump(2 \* (COL / 3), ROW / 2 + 5);

printf("选择错误，请再次选择");

}

}

}

}