**信息与管理科学学院计算机科学系**

实验报告

**课程名称： 人工智能原理及其应用**

**实验名称： 八皇后棋子问题**

**姓 名： 周畅**

**班 级： 计科18-5**

**学 号： 1803101031**

**日 期： 2020/11/16**

**一、实验目的：**

找到所以符合要求的皇后摆放位置。

**二、实验环境**

Dev c++

1. **实验内容**

要求在一个8\*8的棋盘上放置8个皇后，使得它们彼此不受“攻击”。两个

皇后位于棋盘上的同一行、同一列或同一对角线上，则称它们在互相攻击。

现在要找出使得棋盘上8个皇后互不攻击的布局。

**四、实验分析（方法）**

用place[8]数据来保存皇后的位置，

对于上对角线（左上到右下）行-列+7都相等（+7是为了转成正数，方便记录），对于下对角线行+列都相等；

回溯时，当递归到n=7后，回溯到n=6的queen函数中，此时if已经执行完毕，

下一步执行else下的代码// 回溯 flag[i] = 1; d1[n - i + 7] = 1; d2[n + i]= 1;在for循环的作用下，接着寻找第7行是否还有符合条件的位置放queen；向前向后都是如此；

1. **实验步骤（代码等）**

# include <stdio.h>

// 全局变量

int place[8] = { 0 }; // 保存皇后的位置，下标为行，储存的数据为列。

bool flag[8] = { 1, 1, 1, 1, 1, 1, 1, 1 }; // 用来保存哪一列已经存在皇后

bool d1[15] = { 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1 }; // 从左上到右下的对角线为上对角线，每条上对角线上的行和列的差是一样的。

bool d2[15] = { 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1 }; // 从左下到右下的对角线为下对角线，每条下对角线上的行和列的和是一样的。

int key = 0; // 统计解的数量

// 函数声明

void Queen(int); // 八皇后问题

void output(void); // 输出

// 主函数

int main(void)

{

//output();

Queen(0);

return 1;

}

void Queen(int n)

{

for (int i = 0; i < 8; ++i) // 计算第n行皇后的位置

{

if ((flag[i] && d1[n - i + 7]) && d2[n + i]) // 判断第n行第i列的位置是否危险

{

place[n] = i; // 记录皇后的位置

flag[i] = 0; // 记录第i列已经有皇后了

d1[n - i + 7] = 0; // 记录这条上对角线是危险的

d2[n + i] = 0; // 记录这条下对角线是危险的

if (n < 7 ) // 判断八个皇后放完了没有

{

Queen(n + 1);

}

else

{

output(); //输出

}

// 回溯

flag[i] = 1;

d1[n - i + 7] = 1;

d2[n + i]= 1;

}

}

}

void output(void)

{

printf("第%d种解法：\n", ++key);

for (int i = 0; i < 8; ++i)

{

for (int l = 0; l < 8; ++l)

{

if (place[i ]== l)

{

printf("1 ");

}

else

{

printf("0 ");

}

}

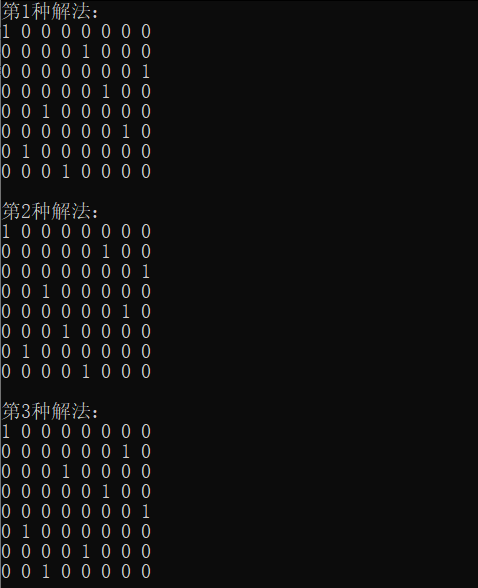
printf("\n");

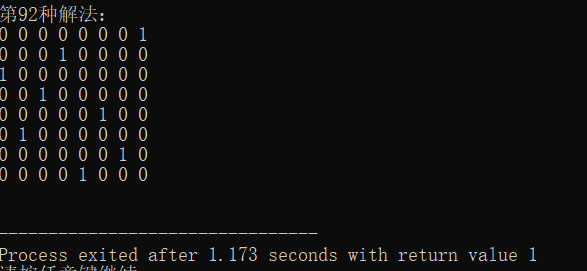
}

printf("\n");

}

**六、实验结果及结论（运行结果截图）**





**七、实验心得**

虽然能达到预期的结果，但在运行时所需的时间比较长，而且总体结构还不够简洁，不太容易去理解。许多问题还需要继续研究，许多技术还需要更多的改进。通过与组内同学的交流，解决了部分问题，今后仍需努力。