武汉大学计算机学院
本科生实验报告

数据结构实验报告
实验二：用堆栈求解 N 皇后问题

专 业 名 称 ： 计算机科学与技术

课 程 名 称 ： 数据结构

指 导 教 师 ： 安　　扬

学 生 学 号 ： 2017301500061

学 生 姓 名 ： 彭 思 翔

学 生 班 级 ： 计科二班

上 机 环 境 ： Visual Studio Code

二〇一八 年 11 月

一、实验题目

实验二：用堆栈求解 N 皇后问题

【问题描述】

请编写一个程序求解 N 皇后问题，在 n*n 的方格盘上，放置 n 个皇后，要求每个皇后的位置不同行，不同列，以及不同对角线，求解及解的数目。

【基本要求】

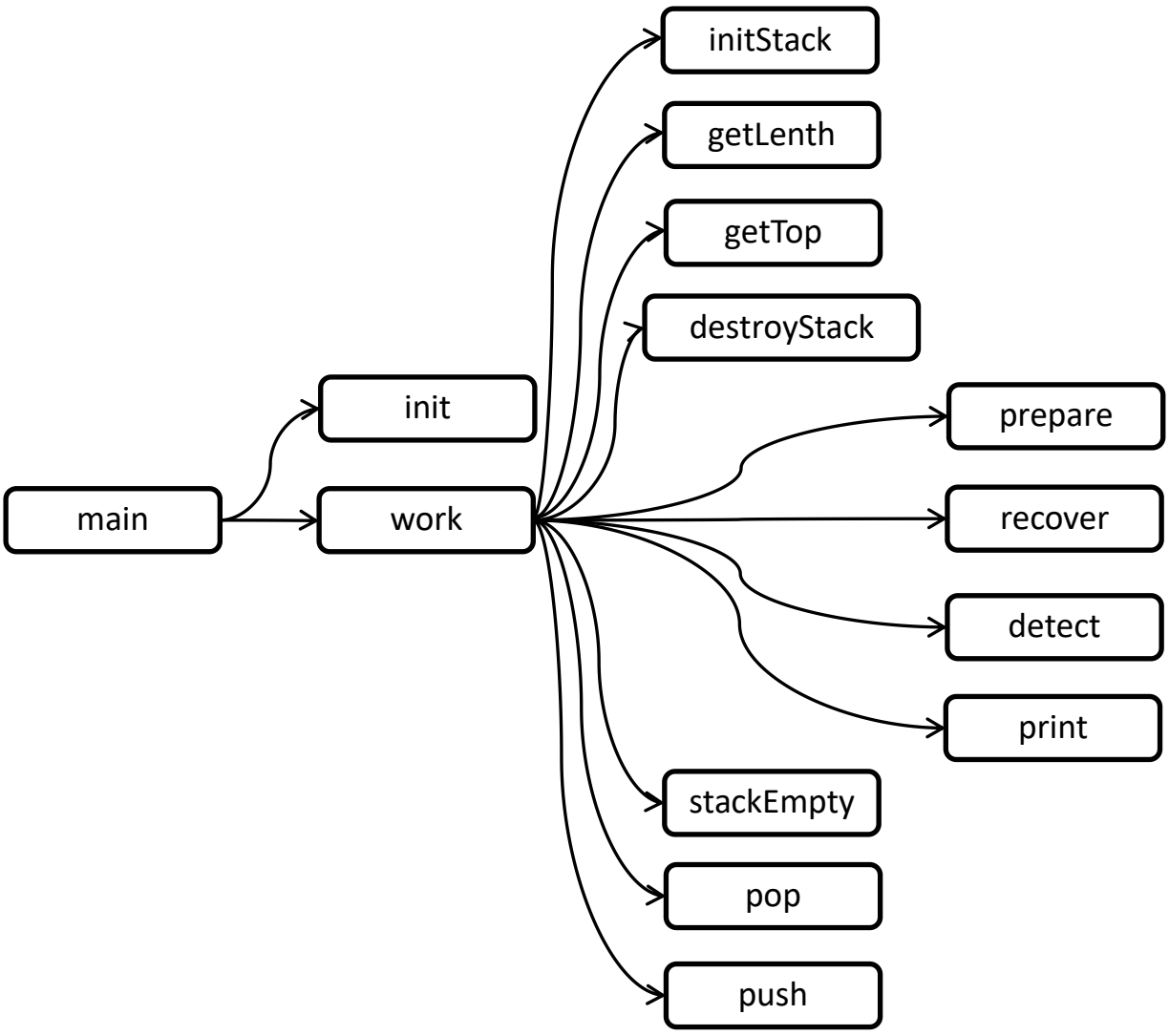
由用户来输入皇后的个数 n，4≤n≤8，要求输出所有的解。

二、实验项目的目的

掌握栈的实现和运用，顺便与经典的用递归来实现的八皇后问题比较有什么不同，尝试对 N 皇后的解法进行优化。

三、实验项目程序结构

Main 函数　　面向过程函数　　栈实现函数　　算法功能函数



四、实验项目中各文件函数功能描述

```
void init();                    //输入皇后个数 n
void work();                    //N 皇后求解算法实现
bool pop(Stack*&);              //弹出栈顶元素
void print(Stack*&);            //根据栈打印当前解
int getTop(Stack*&);            //获取栈顶元素
int getLenth(Stack*&);          //获取栈长度
bool detect(int, int);          //前一个参数为行，后一参数为列，用标记检测当前位置是否可以放皇后
void prepare(int, int);         //前一个参数为行，后一参数为列，在当前位置放皇后，设置标记
void recover(int, int);         //前一个参数为行，后一参数为列，移除当前位置皇后，恢复标记
bool push(Stack*&, int);        //入栈
void initStack(Stack*&);        //新建栈并初始化
bool stackEmpty(Stack*&);       //判断栈是否为空，为空返回 1，不为空返回 0
void destroyStack(Stack*&);     //销毁栈
```

五、算法描述

【数据结构】

栈：栈自底向上表示行，栈所存储的值表示该行所放皇后的列的位置。

```
typedef struct {
int data[MaxSize];
int top;
}Stack;
```

【设计思路】

（1）皇后合法性判断：在判断某行某列(x, y)是否可以放八皇后时，对于行，栈的位置不同则行不同；对于列，用数组 col[N]作为标记表示第 i 列是否有皇后；对于左斜线，从下至上依次编号 1~2n-1，(x, y)位于编号 y-x+n 的左对角线，用数组 leftDia[2N-1]作为标记表示第 i 个左斜线是否有皇后;对于右斜线,从下至上依次编号 1~2n-1,(x, y)位于编号 n×2-x-y-1 的右对角线，用数组 rightDia[2N-1]作为标记表示第 i 个右斜线是否有皇后。

1	2	…	N-1	N
1	2	…	N-1	N
1	2	…	N-1	N
1	2	…	N-1	N
1	2	…	N-1	N

列标记编号

…	2N-4	2N-3	2N-2	2N-1
4	…	2N-4	2N-3	2N-2
3	4	…	2N-4	2N-3
2	3	4	…	2N-4
1	2	3	4	…

左斜线标记编号

2N-1	2N-2	2N-3	2N-4	…
2N-2	2N-3	2N-4	…	4
2N-3	2N-4	…	4	3
2N-4	…	4	3	2
…	4	3	2	1

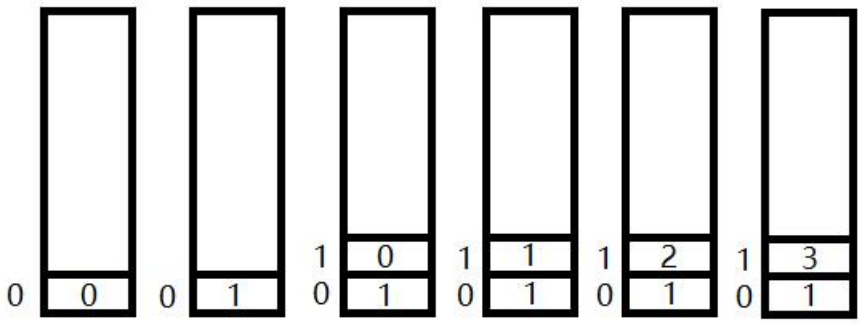
右斜线标记编号

时间复杂度 O(1)

```
bool detect(int x, int y) {
    if (col[y], leftDia[y - x + n], rightDia[(n << 1) - y - x + 1]都没有标记过)
        return true;
```

```
        return false;
    }
```

（2）非递归求解实现：初始在栈内放入(0, 0)表示第 1 行第 0 列放八皇后作为栈初始化。在循环中，获取栈长度为 x，取出栈顶元素作为 y，由(x, y+1)至(x, n)枚举循环，依次判断合法性，如果(x, i)可以放八皇后，i∈[y+1,n]，则栈顶元素改为 i，并新加入栈元素 0，表示(x+1, 0)放入八皇后做预备。i 枚举至 n 以后栈顶出栈，表示返回上一行。如果 x==n 且找到了解，则输出。



时间复杂度小于 O(n!)

```
void work() {
    初始化栈 s;
    push(s, 0);
    while(s 为空) {
        int x = s 的长度, y = s 的栈顶元素;
        弹出 s 的栈顶元素并消除标记;
        for (i→y+1,n)
            if (在(x, i)处放置皇后合法) {
                设置(x, i)处标记，并加入栈;
                x==n 则输出，否则 push(s, 0);
                break;
            }
    }
    销毁栈并输出总数;
}
```

六、实验数据和实验结果分析

运行结果良好。

```
6
 . × . . . .
 . . . × . .
 . . . . . ×
 × . . . . .
 . . × . . .
 . . . . × .

 . . × . . .
 . . . . . ×
 . × . . . .
 . . . . × .
 × . . . . .
 . . . × . .

 . . . × . .
 × . . . . .
 . . . . × .
 . × . . . .
 . . . . . ×
 . . × . . .

 . . . . × .
 . . × . . .
 × . . . . .
 . . . . . ×
 . . . × . .
 . × . . . .

Total: 4
请按任意键继续. . .
```

七、实验体会

我们都知道八皇后问题是个有趣的经典问题，解法往往从递归开始讲起，所以这次采用栈我就尝试思考递归到非递归的转化。学了计组就知道，系统在实现递归的时候本质也是使用了系统栈的。所以这次在进行算法设计的时候就想尝试模拟系统栈实现递归的方式实现八皇后。最终结果也很好。另外在网上看见了疑似用构造法 O(1)实现八皇后解的方法，有机会可以研究一下。