实验6 回溯法与分支限界法

# 实验要求

1 掌握回溯法求解问题的思想，学会利用其原理求解相关问题。

2 理解分支限界算法的优先搜寻原理及一般应用。

3 编程实现典型分支限界算法，理解“分支”、“限界”思想，并对算法进行验证分析。

# 二. 实验内容

1. **八皇后问题**

八皇后问题是一个古老而著名的问题，是回溯算法的典型例题。该问题是十九世纪著名的数学家高斯1850年提出：在8×8格的国际象棋上摆放八个皇后，使其不能互相攻击，即任意两个皇后都不能处于同一行、同一列或同一斜线上，问有多少种摆法。

高斯认为有76种方案。1854年在柏林的象棋杂志上不同的作者发表了40种不同的解，后来有人用图论的方法解出92种结果。

要求编程实现回溯法，并进行验证，并使其能扩展到任意的皇后数的情况，同时对源程序给出详细的注释。

**2. 分支限界法0-1背包问题**

示例输入（规定物品数量为10，背包容量为50，输入为20个数，**前十个为物品重量，后十个数为物品价值**）：

**12  
3  
11  
5  
6  
8  
9  
4  
7  
10**6  
2  
7  
3  
2  
9  
8  
10  
4  
5  
示例输出（**最大价值**）：

44

**三、实验步骤**

1. 理解算法思想和问题要求；

2. 编程实现题目要求

3. 上机输入和调试自己所编的程序；

4. 验证分析实验结果；

5. 整理出实验报告。

**四、实验结果**

分支限界法0-1背包问题

/\*

\*20140031 陈小龙 计科2班

\*/

#include<iostream>

#include<queue>

**using** **namespace** std**;**

**typedef** struct treenode**{**

int weight**;**

int value**;**

int level**;**

int flag**;**

**}**treenode**;**

queue**<**struct treenode**>** que**;**

int enQueue**(**int w**,**int v**,**int level**,**int flag**,**int n**,**int**\*** bestvalue**)**

**{**

treenode node**;**

node**.**weight **=** w**;**

node**.**value **=** v**;**

node**.**level **=** level**;**

node**.**flag **=** flag**;**

**if** **(**level **==** n**)**

**{**

**if** **(**node**.**value **>** **\***bestvalue**)**

**{**

**\***bestvalue **=** node**.**value**;**

**}**

**return** 0**;**

**}else**

**{**

que**.**push**(**node**);**

**}**

**}**

//w为重量数组，v为价值数组，n为物品个数，c为背包容量，bestValue为全局最大价值

int bbfifoknap**(**int w**[],**int v**[],**int n**,**int c**,**int**\*** bestValue**)**

**{**

//初始化结点

int i**=**1**;**

treenode tag**,** livenode**;**

livenode**.**weight **=** 0**;**

livenode**.**value **=** 0**;**

livenode**.**level **=** 1**;**

livenode**.**flag **=** 0**;**//初始为0

tag**.**weight **=** **-**1**;**

tag**.**value **=** 0**;**

tag**.**level **=** 0**;**

tag**.**flag **=** 0**;**//初始为0

que**.**push**(**tag**);**

**while** **(**1**)**

**{**

**if** **(**livenode**.**weight **+** w**[**i**-**1**]** **<=** c**)**

**{**

enQueue**(**livenode**.**weight **+** w**[**i**-**1**],** livenode**.**value **+** v**[**i**-**1**],** i**,** 1**,**n**,**bestValue**);**

**}**

enQueue**(**livenode**.**weight**,**livenode**.**value**,** i**,** 0**,**n**,**bestValue**);**

livenode **=** que**.**front**();**

que**.**pop**();**

**if** **(**livenode**.**weight **==** **-**1**)**

**{**

**if** **(**que**.**empty**()** **==** 1**)**

**{**

**break;**

**}**

livenode **=** que**.**front**();**

que**.**pop**();**

que**.**push**(**tag**);**

i**++;**

**}**

**}**

**return** 0**;**

**}**

int main**()**

**{**

int w**[]** **=** **{**12**,**3**,**11**,**5**,**6**,**8**,**9**,**4**,**7**,**10**};**

int v**[]** **=** **{**6**,**2**,**7**,**3**,**2**,**9**,**8**,**10**,**4**,**5**};**

int n **=** 10**;**

int c **=** 50**;**

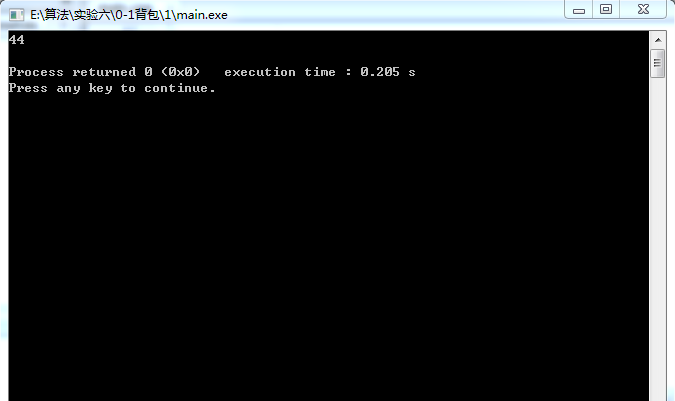
int bestValue**=**0**;**

bbfifoknap**(**w**,**v**,**n**,**c**,&**bestValue**);**

cout **<<** bestValue**<<**endl**;**

**return** 0**;**

**}**



八皇后问题

/\*

\*20140031 陈小龙 计科2班

\*/

#include<stdio.h>

#include<stdlib.h>

#include<math.h>

int x**[**100**];**

bool place**(**int k**)** //考察皇后k放置在x[k]列是否发生冲突

**{**

int i**;**

**for(**i**=**1**;**i**<**k**;**i**++)**

**if(**x**[**k**]==**x**[**i**]||**abs**(**k**-**i**)==**abs**(**x**[**k**]-**x**[**i**]))**

**return** **false;**

**return** **true;**

**}**

void queue**(**int n**)**

**{**

int i**,**k**;**

**for(**i**=**1**;**i**<=**n**;**i**++)**

x**[**i**]=**0**;**

k**=**1**;**

**while(**k**>=**1**)**

**{**

x**[**k**]=**x**[**k**]+**1**;** //在下一列放置第k个皇后

**while(**x**[**k**]<=**n**&&!**place**(**k**))**

x**[**k**]=**x**[**k**]+**1**;** //搜索下一列

**if(**x**[**k**]<=**n**&&**k**==**n**)** //得到一个输出

**{**

**for(**i**=**1**;**i**<=**n**;**i**++)**

printf**(**"%d "**,**x**[**i**]);**

printf**(**"\n"**);**

//若return则只求出其中一种解，若不return则可以继续回溯，求出全部的可能的解

//return;

**}**

**else** **if(**x**[**k**]<=**n**&&**k**<**n**)**

k**=**k**+**1**;** //放置下一个皇后

**else**

**{**

x**[**k**]=**0**;** //重置x[k],回溯

k**=**k**-**1**;**

**}**

**}**

**}**

int main**()**

**{**

int n**;**

printf**(**"输入皇后个数n:"**);**

scanf**(**"%d"**,&**n**);**

queue**(**n**);**

**return** 0**;**

**}**

