目录

一、实验题目

1、0/1 Knapsack Problem. There are 5 items that have a value and weight list below, the knapsack can contain at most 100 Lbs. Solve the problem using back-tracking algorithm and try to draw the tree generated

2、Solve the 8-Queen problem using back-tracking algorithm.

二、实验目的

理解回溯法的思想，掌握使用回溯法解决问题的能力，能够 shi 用回溯法解决经典问题，例如 0/1 背包和八皇后问题等

三、实验设计与分析

1、0/1 Knapsack Problem

对于 0/1 背包问题，每个物品来说，只有要或者不要，即 0 或 1 的问题，所以我们可以先列出一个二叉树，将性价比最大值放在解空间树的根结点处，从根结点处开始进行深度优先遍历。中间过程因为有约束函数，所以不会一直深度遍历下去，如果 k(k<n)号物品加入后超出了背包容量时,取消选择 k,回溯至最近的加入背包的物品，选择另一条分支继续 DFS。在这个过程中使用 bond()函数进行剪枝。

2、Solve the 8-Queen problem

使用递归地方法，如果当前行存在可以放置皇后的位置就放置，然后递归至下一行寻找合适的位置，当皇后全部放完时输出结果。

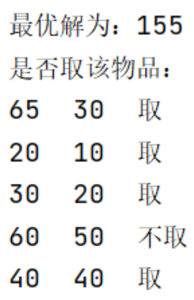
四、实验环境

Windows 10、IntelliJ IDEA 2020.3.2 x64、jdk1.8.0_281

五、项目测试（功能和性能）

1、0/1 Knapsack Problem

实验结果如下图

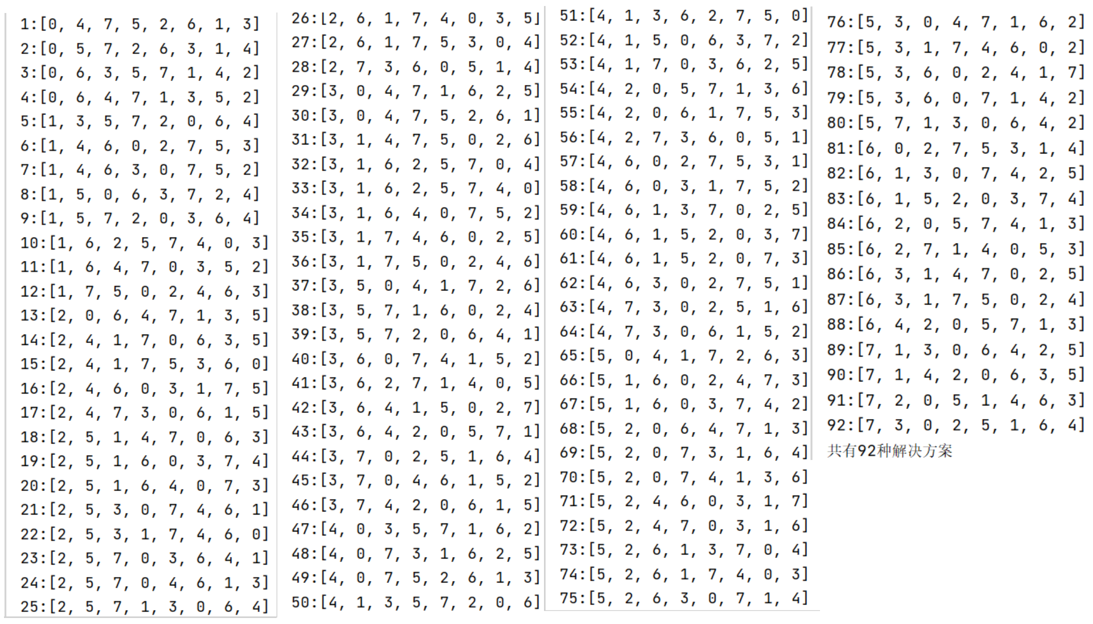


```
最优解为：155
是否取该物品：
65   30   取
20   10   取
30   20   取
60   50   不取
40   40   取
```

算法时间复杂度为 0(2^n)

2、Solve the 8-Queen problem

实验结果如下图，算法时间复杂度为 0(n!)



```
1:[0, 4, 7, 5, 2, 6, 1, 3]
2:[0, 5, 7, 2, 6, 3, 1, 4]
3:[0, 6, 3, 5, 7, 1, 4, 2]
4:[0, 6, 4, 7, 1, 3, 5, 2]
5:[1, 3, 5, 7, 2, 0, 6, 4]
6:[1, 4, 6, 0, 2, 7, 5, 3]
7:[1, 4, 6, 3, 0, 7, 5, 2]
8:[1, 5, 0, 6, 3, 7, 2, 4]
9:[1, 5, 7, 2, 0, 3, 6, 4]
10:[1, 6, 2, 5, 7, 4, 0, 3]
11:[1, 6, 4, 7, 0, 3, 5, 2]
12:[1, 7, 5, 0, 2, 4, 6, 3]
13:[2, 0, 6, 4, 7, 1, 3, 5]
14:[2, 4, 1, 7, 0, 6, 3, 5]
15:[2, 4, 1, 7, 5, 3, 6, 0]
16:[2, 4, 6, 0, 3, 1, 7, 5]
17:[2, 4, 7, 3, 0, 6, 1, 5]
18:[2, 5, 1, 4, 7, 0, 6, 3]
19:[2, 5, 1, 6, 0, 3, 7, 4]
20:[2, 5, 1, 6, 4, 0, 7, 3]
21:[2, 5, 3, 0, 7, 4, 6, 1]
22:[2, 5, 3, 1, 7, 4, 6, 0]
23:[2, 5, 7, 0, 3, 6, 4, 1]
24:[2, 5, 7, 0, 4, 6, 1, 3]
25:[2, 5, 7, 1, 3, 0, 6, 4]
```

```
26:[2, 6, 1, 7, 4, 0, 3, 5]
27:[2, 6, 1, 7, 5, 3, 0, 4]
28:[2, 7, 3, 6, 0, 5, 1, 4]
29:[3, 0, 4, 7, 1, 6, 2, 5]
30:[3, 0, 4, 7, 5, 2, 6, 1]
31:[3, 1, 4, 7, 5, 0, 2, 6]
32:[3, 1, 6, 2, 5, 7, 0, 4]
33:[3, 1, 6, 2, 5, 7, 4, 0]
34:[3, 1, 6, 4, 0, 7, 5, 2]
35:[3, 1, 7, 4, 6, 0, 2, 5]
36:[3, 1, 7, 5, 0, 2, 4, 6]
37:[3, 5, 0, 4, 1, 7, 2, 6]
38:[3, 5, 7, 1, 6, 0, 2, 4]
39:[3, 5, 7, 2, 0, 6, 4, 1]
40:[3, 6, 0, 7, 4, 1, 5, 2]
41:[3, 6, 2, 7, 1, 4, 0, 5]
42:[3, 6, 4, 1, 5, 0, 2, 7]
43:[3, 6, 4, 2, 0, 5, 7, 1]
44:[3, 7, 0, 2, 5, 1, 6, 4]
45:[3, 7, 0, 4, 6, 1, 5, 2]
46:[3, 7, 4, 2, 0, 6, 1, 5]
47:[4, 0, 3, 5, 7, 1, 6, 2]
48:[4, 0, 7, 3, 1, 6, 2, 5]
49:[4, 0, 7, 5, 2, 6, 1, 3]
50:[4, 1, 3, 5, 7, 2, 0, 6]
```

```
51:[4, 1, 3, 6, 2, 7, 5, 0]
52:[4, 1, 5, 0, 6, 3, 7, 2]
53:[4, 1, 7, 0, 3, 6, 2, 5]
54:[4, 2, 0, 5, 7, 1, 3, 6]
55:[4, 2, 0, 6, 1, 7, 5, 3]
56:[4, 2, 7, 3, 6, 0, 5, 1]
57:[4, 6, 0, 2, 7, 5, 3, 1]
58:[4, 6, 0, 3, 1, 7, 5, 2]
59:[4, 6, 1, 3, 7, 0, 2, 5]
60:[4, 6, 1, 5, 2, 0, 3, 7]
61:[4, 6, 1, 5, 2, 0, 7, 3]
62:[4, 6, 3, 0, 2, 7, 5, 1]
63:[4, 7, 3, 0, 2, 5, 1, 6]
64:[4, 7, 3, 0, 6, 1, 5, 2]
65:[5, 0, 4, 1, 7, 2, 6, 3]
66:[5, 1, 6, 0, 2, 4, 7, 3]
67:[5, 1, 6, 0, 3, 7, 4, 2]
68:[5, 2, 0, 6, 4, 7, 1, 3]
69:[5, 2, 0, 7, 3, 1, 6, 4]
70:[5, 2, 0, 7, 4, 1, 3, 6]
71:[5, 2, 4, 6, 0, 3, 1, 7]
72:[5, 2, 4, 7, 0, 3, 1, 6]
73:[5, 2, 6, 1, 3, 7, 0, 4]
74:[5, 2, 6, 1, 7, 4, 0, 3]
75:[5, 2, 6, 3, 0, 7, 1, 4]
```

```
76:[5, 3, 0, 4, 7, 1, 6, 2]
77:[5, 3, 1, 7, 4, 6, 0, 2]
78:[5, 3, 6, 0, 2, 4, 1, 7]
79:[5, 3, 6, 0, 7, 1, 4, 2]
80:[5, 7, 1, 3, 0, 6, 4, 2]
81:[6, 0, 2, 7, 5, 3, 1, 4]
82:[6, 1, 3, 0, 7, 4, 2, 5]
83:[6, 1, 5, 2, 0, 3, 7, 4]
84:[6, 2, 0, 5, 7, 4, 1, 3]
85:[6, 2, 7, 1, 4, 0, 5, 3]
86:[6, 3, 1, 4, 7, 0, 2, 5]
87:[6, 3, 1, 7, 5, 0, 2, 4]
88:[6, 4, 2, 0, 5, 7, 1, 3]
89:[7, 1, 3, 0, 6, 4, 2, 5]
90:[7, 1, 4, 2, 0, 6, 3, 5]
91:[7, 2, 0, 5, 1, 4, 6, 3]
92:[7, 3, 0, 2, 5, 1, 6, 4]
共有92种解决方案
```