*# Assignment #6: Recursion and DP*

付耀贤 信息管理系 2400016634

*## 1. 题目*

*### sy119: 汉诺塔*

recursion, https://sunnywhy.com/sfbj/4/3/119

思路：

没有思路，向AI初步学习递归的写法，好绕。。。

代码：

def hanoi(n, source, target, mid):

    if n == 1:

        print(f"{source}->{target}")

    else:

        # 1. 将 n-1 个盘子从源柱子移动到辅助柱子

        hanoi(n-1, source, mid, target)

        # 2. 将第 n 个盘子从源柱子移动到目标柱子

        print(f"{source}->{target}")

        # 3. 将 n-1 个盘子从辅助柱子移动到目标柱子

        hanoi(n-1, mid, target, source)

n = int(input())

print(2\*\*n - 1)

hanoi(n, 'A', 'C', 'B')

代码运行截图 <mark>（至少包含有"Accepted"）</mark>



*### sy132: 全排列I*

recursion, https://sunnywhy.com/sfbj/4/3/132

思路：

看到这个题心中大喜：使用itertools即可；后来又把dfs代码看了一遍，得出结论：和汉诺塔一样晕晕乎乎。

代码：

import itertools  
n = int(input())  
perm = list(itertools.permutations(range(1,n+1)))  
for i in perm:  
 print(\*i)

def dfs(idx, n, used, temp, result):  
 if idx == n + 1:   
 result.append(temp[:])   
 return  
 for i in range(1, n + 1):  
 if not used[i]:   
 temp.append(i)   
 used[i] = True   
 dfs(idx + 1, n, used, temp, result) # 递归处理下一个位置  
 used[i] = False # 回溯：将数字 i 标记为未使用  
 temp.pop() # 回溯：将数字 i 从当前排列中移除  
  
def generate\_permutations(n):  
 result = []  
 used = [False] \* (n + 1)   
 dfs(1, n, used, [], result)   
 for perm in result:  
 print(" ".join(map(str, perm)))   
   
n = int(input())  
generate\_permutations(n)

代码运行截图 ==（至少包含有"Accepted"）==



*### 02945: 拦截导弹*

dp, http://cs101.openjudge.cn/2024fallroutine/02945

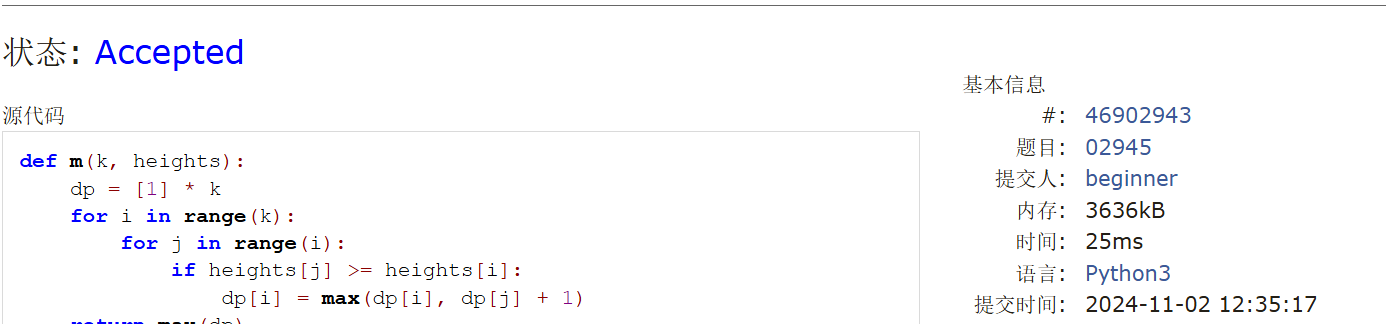
思路：

为数不多我还有思路的题目，主要是更新dp数组，和以前做的几道题很类似

代码：

def m(k, heights):  
 dp = [1] \* k  
 for i in range(k):  
 for j in range(i):  
 if heights[j] >= heights[i]:  
 dp[i] = max(dp[i], dp[j] + 1)  
 return max(dp)  
  
k = int(input())  
heights = list(map(int, input().split()))  
result = m(k, heights)  
print(result)

代码运行截图 <mark>（至少包含有"Accepted"）</mark>



*### 23421: 小偷背包*

dp, http://cs101.openjudge.cn/practice/23421

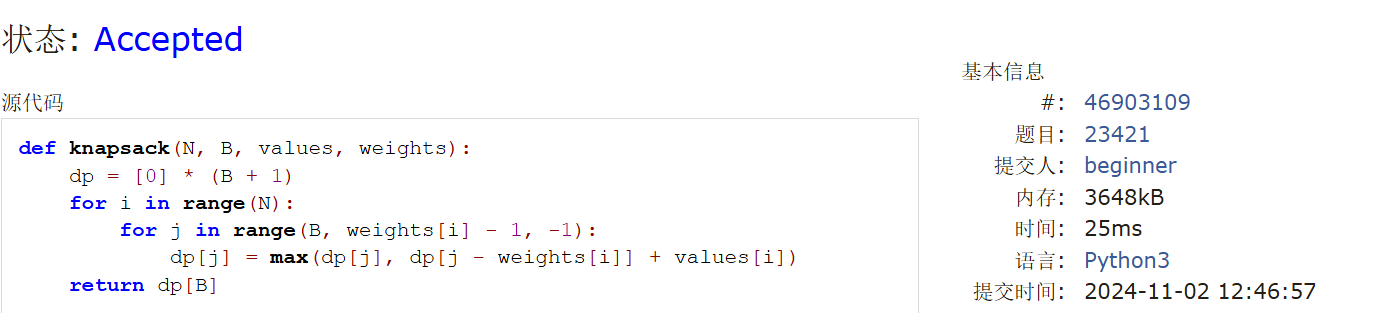
思路：

这个dp比上一个绕了，已经写不出来了。。需要AI帮助

代码：

def knapsack(N, B, values, weights):  
 dp = [0] \* (B + 1)  
 for i in range(N):  
 for j in range(B, weights[i] - 1, -1):  
 dp[j] = max(dp[j], dp[j - weights[i]] + values[i])  
 return dp[B]  
  
N, B = map(int, input().split())  
values = list(map(int, input().split()))  
weights = list(map(int, input().split()))  
result = knapsack(N, B, values, weights)  
print(result)

代码运行截图 <mark>（至少包含有"Accepted"）</mark>



*### 02754: 八皇后*

dfs and similar, http://cs101.openjudge.cn/practice/02754

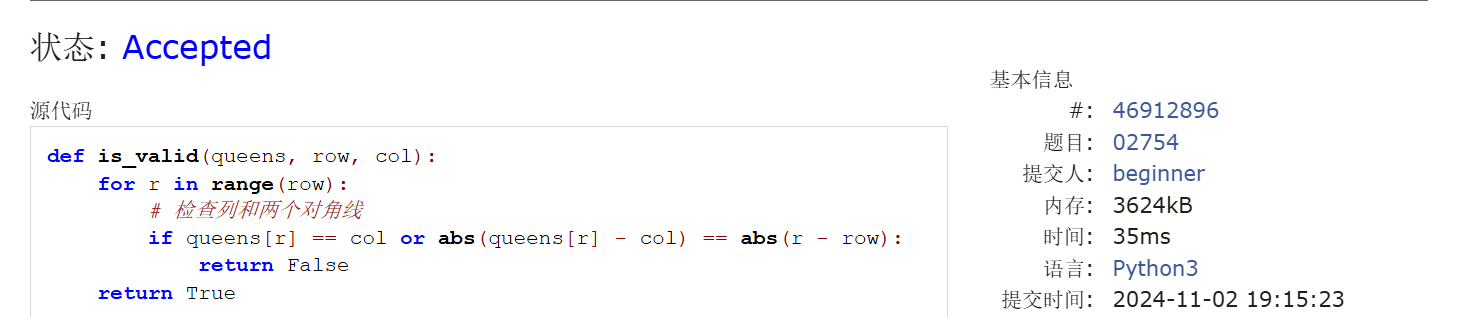
思路：

没有思路，全靠AI来学习，虽然还是半懂不懂。。

代码：

def is\_valid(queens, row, col):  
 for r in range(row):  
 # 检查列和两个对角线  
 if queens[r] == col or abs(queens[r] - col) == abs(r - row):  
 return False  
 return True  
  
def solve\_n\_queens(row, queens, solutions):  
 if row == 8: # 找到一个解  
 solutions.append(queens[:])  
 return  
 for col in range(8):  
 if is\_valid(queens, row, col):  
 queens[row] = col  
 solve\_n\_queens(row + 1, queens, solutions)  
  
solutions = []  
solve\_n\_queens(0, [-1]\*8, solutions)  
  
queen\_strings = [''.join(str(col + 1) for col in solution) for solution in solutions]  
  
n = int(input())  
results = []  
for \_ in range(n):  
 b = int(input())  
 # 输出对应的第b个皇后串（-1因为b从1开始）  
 results.append(queen\_strings[b - 1])  
print("\n".join(results))

代码运行截图 <mark>（至少包含有"Accepted"）</mark>



*### 189A. Cut Ribbon*

brute force, dp 1300 https://codeforces.com/problemset/problem/189/A

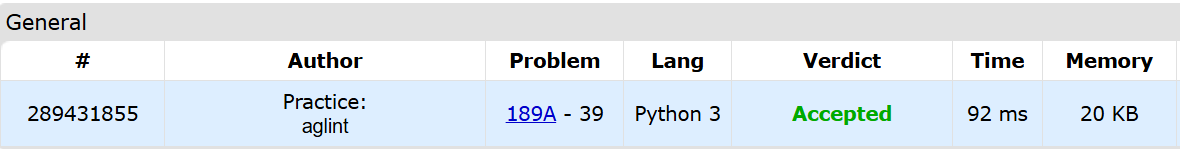
思路：

这是一道我看AI能大概看明白的题，点赞！

代码：

def max\_cuttings(n, a, b, c):  
 dp = [-1] \* (n + 1)  
 dp[0] = 0  
 for i in range(1, n + 1):  
 if i >= a and dp[i - a] != -1:  
 dp[i] = max(dp[i], dp[i - a] + 1)  
 if i >= b and dp[i - b] != -1:  
 dp[i] = max(dp[i], dp[i - b] + 1)  
 if i >= c and dp[i - c] != -1:  
 dp[i] = max(dp[i], dp[i - c] + 1)  
 return dp[n]  
  
n, a, b, c = map(int, input().split())  
result = max\_cuttings(n, a, b, c)  
print(result)

代码运行截图 <mark>（至少包含有"Accepted"）</mark>



*## 2. 学习总结和收获*

DP类题目已经可以看懂且有思路了，比前几周进步很大！递归 、DFS脑子是一片空白、一团乱麻，过了期中再好好琢磨琢磨，争取能理解。这周的题目太难了，基本都写不出来，全靠AI解答，而且AI详细解释以后我还是觉得很难理解QAQ。