Assignment #6: Recursion and DP

Updated 2201 GMT+8 Oct 29, 2024

2024 fall, Complied by <mark>颜鼎堃 工学院</mark>

说明:

- 1)请把每个题目解题思路(可选),源码Python,或者C++(已经在Codeforces/Openjudge上AC),截图(包含Accepted),填写到下面作业模版中(推荐使用 typora https://typoraio.cn ,或者用word)。AC 或者没有AC,都请标上每个题目大致花费时间。
- 3)提交时候先提交pdf文件,再把md或者doc文件上传到右侧"作业评论"。Canvas需要有同学清晰头像、提交文件有pdf、"作业评论"区有上传的md或者doc附件。
- 4) 如果不能在截止前提交作业,请写明原因。

1. 题目

sy119: 汉诺塔

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

思路:

- 递归基础练习
- 填参数的时候在草稿纸上画一画移动方法

代码:

```
Python

def Hannoi(n, stt, wth, end):
    if n = 1:
        print(f"{stt}→{end}")
    else:
        Hannoi(n-1, stt, end, wth)
        print(f"{stt}→{end}")
        Hannoi(n-1, wth, stt, end)

def main():
    n = int(input())
    print(2**n - 1)
    Hannoi(n, "A", "B", "C")

main()
```

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

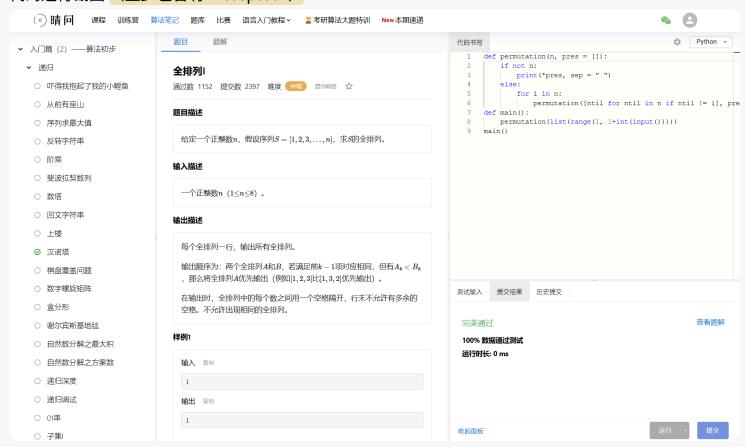
思路:

- 递归的第二个练习
- 最开始编写的时候,每一行只输出和上一行不一样的部分,连每行的长度都不一致,我选择将输出 改为添加到列表,再统一修改,但是调用全局变量消耗大量时间,在 n=8 时超时
- 后来想到可以在传递参数时额外传递一项,但在每次传递前都对变量重新赋值,导致结果还是对不上,重新进行学习后发现只需改变传递时的参数表达式而不需要直接赋值,最终通过

```
Python

def permutation(n, pres = []):
    if not n:
        print(*pres, sep = " ")
    else:
        for i in n:
            permutation([ntil for ntil in n if ntil ≠ i], pres+[i])

def main():
    permutation(list(range(1, 1+int(input()))))
main()
```



02945: 拦截导弹

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

思路:

- 即给定一个数列,求最长单调递减子列
- 学习了一下动态规划的基本知识,这题的思路(和下题)基本都是学的Ol Wiki 动态规划基础 Ol Wiki

当然,本题还要感谢信科同学的帮助

- 基本思路: 从后往前,数每个数后面的子列并把自己放在最长的一个的最前面
- 按照OI Wiki,复杂度为 $O(n^2)$,不是最佳算法,但完全能用

```
Python

k = int(input())
mis = list(map(int, input().split()))
length = [0 for i in range(k)]
length[k-1] = 1
for i in range(k-2, -1, -1):
    maximum = 0
    for j in range(i, k):
        if mis[j] ≤ mis[i]:
            maximum = max(maximum, length[j])
length[i] = maximum + 1
```

```
print(max(length))
```



23421: 小偷背包

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

思路:

学习的OI Wiki

背包 DP - OI Wiki

• 转移方程: $f(j) = max(f(j), f(j-w_i) + v_i)$, 运用二重循环

```
Python

N, B = map(int, input().split())
p = list(map(int, input().split()))
w = list(map(int, input().split()))
maximum = [0 for j in range(B+1)]
for i in range(N):
    for j in range(B, w[i]-1, -1):
        maximum[j] = max(maximum[j], maximum[j-w[i]] + p[i])
print(max(maximum))
```



02754: 八皇后

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

思路:

- 感觉比全排列简单,没借助外力自己解决了
- 先递归打表,再直接查找(注释部分为打表代码)
- 递归思路: n 为行数,从 0 到 7 ,将已经放好了的皇后的位置存入 pres 列表,在一行中对各列进行判断,去除不合题意的位置(即已有的列和位于同一斜线的位置),然后进行下一次递归,到 n=8 时将 pres 作为答案,递归结束

```
queen = ['15863724', '16837425', '17468253', '17582463', '24683175', '25713864',
'25741863', '26174835', '26831475', '27368514', '27581463', '28613574', '31758246',
'35281746', '35286471', '35714286', '35841726', '36258174', '36271485', '36275184',
'36418572', '36428571', '36814752', '36815724', '36824175', '37285146', '37286415',
'38471625', '41582736', '41586372', '42586137', '42736815', '42736851', '42751863',
'42857136', '42861357', '46152837', '46827135', '46831752', '47185263', '47382516',
'47526138', '47531682', '48136275', '48157263', '48531726', '51468273', '51842736',
'53847162', '57138642', '57142863', '57248136', '57263148', '57263184', '57413862',
'58413627', '58417263', '61528374', '62713584', '62714853', '63175824', '63184275',
'63185247', '63571428', '63581427', '63724815', '63728514', '63741825', '64158273',
'64285713', '64713528', '64718253', '68241753', '71386425', '72418536', '72631485',
'73168524', '73825164', '74258136', '74286135', '75316824', '82417536', '82531746',
```

```
'83162574', '84136275']
# def eightqueens(n, pres):
    if n = 8:
        global queen
#
        queen.append("".join(list(map(lambda t: str(t+1), pres))))
    else:
#
#
        for i in range(8):
            for j in range(n):
                if i = pres[j] or n - i = j - pres[j] or n + i = j + pres[j]:
#
#
                    break
#
            else:
#
                eightqueens(n+1, pres+[i])
# eightqueens(0, [])
# print(queen)
for i in range(int(input())):
    print(queen[int(input())-1])
```



189A. Cut Ribbon

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

思路:

- 结合前两个学习了他人思路的题目,独立完成了本题
- 感觉似乎和完全背包问题有点像,相当于三种物品装袋子,每个物品数量无限;但事实上有着重要的区别,即袋子必须要装满(丝带最终长度归零),这导致完全背包问题的思路在此并不适用

- 事实上,本题与一道经典的小学奥数题有相似之处:一共八层楼梯,每次上一层或者两层,一共有多少种上楼方法(即斐波那契数列),但本题求的不是剪丝带方法,而是丝带段数,但都可以由初始状态递推得到
- 对应的方程:

$$f(n) = \max_{i=1,2,3,a[i]
eq 0} \left(f\left(n-a\left[i
ight]
ight) + 1, f(n)
ight)$$

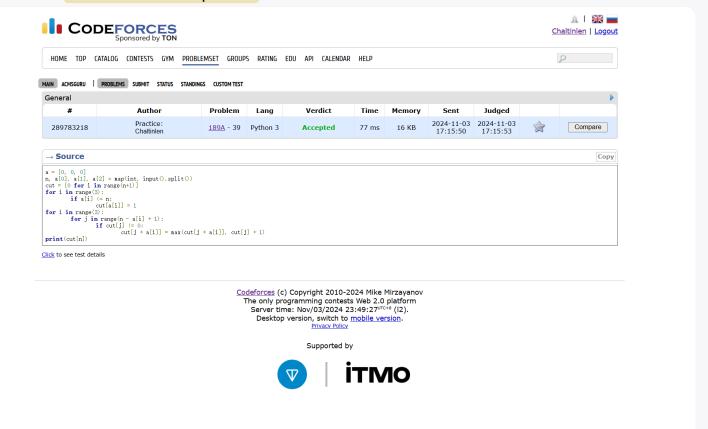
代码:

```
Python

a = [0, 0, 0]
n, a[0], a[1], a[2] = map(int, input().split())
cut = [0 for i in range(n+1)]
for i in range(3):
    if a[i] ≤ n:
        cut[a[i]] = 1

for i in range(3):
    for j in range(n - a[i] + 1):
        if cut[j] ≠ 0:
        cut[j + a[i]] = max(cut[j + a[i]], cut[j] + 1)
print(cut[n])
```

代码运行截图 (至少包含有"Accepted")



2. 学习总结和收获

如果作业题目简单,有否额外练习题目,比如:OJ"计概2024fall每日选做"、CF、LeetCode、洛谷等网站题目。

- 1. 真心希望老师上课讲点例题,只给大方向的话,面对具体题目还是得靠自学
- 2. AI是人工智障我是智障人工,每天看群里怎么那么多同学反映题目简单,感觉我可以和我的期中考试成绩说再见了
- 3. 菜就多练