

Updated 2201 GMT+8 Oct 29, 2024

2024 fall, Compiled by <mark>同学的姓名、院系</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: 递归问题, 先将前 $n-1$ 个盘子从 `start` 借助 `goal` 移到 `middle`, 再将 $n-1$ 个盘子从 `middle` 借助 `start` 移动到 `goal`, 易知移动的最小次数一定是 $2^n - 1$, 最后使用递归函数即可。(1h)

代码:

```
def moveHanoi(n, start, goal, middle):  
    if n == 0:  
        return  
    moveHanoi(n - 1, start, middle, goal)  
    print(f"{from_rod}->{to_rod}")  
    moveHanoi(n - 1, middle, goal, start)  
  
n = int(input())  
print(2**n - 1)  
moveHanoi(n, 'A', 'C', 'B')
```

代码运行截图 <mark>（至少包含有"Accepted"）</mark>

完美通过

100% 数据通过测试

运行时长: 0 ms

sy132: 全排列 I

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

思路：调用了两个函数，第一个函数用于处理 1 到 n 中的每一个数以及当当前位置大于 n 后生成新列表的问题，第二个函数则用于初始化数组及生成排列，并最后按要求输出序列即可。（1h）

代码:

```
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)  
            used[i] = False  
            temp.pop()  
  
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"）==

测试输入

提交结果

历史提交

完美通过

100% 数据通过测试

运行时长: 0 ms

收起面板

运行

提交

02945：拦截导弹

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

思路: 关键部分, 先创建一个长度为 k 的列表并将数据全部初始为 1, 然后通过两层循环来填充 `dp` 列表, 最后通过 `max` 函数来找出 `dp` 列表中的最大值即可。(1.5h)

代码:

```
k = int(input())  
missile_heights = list(map(int, input().split()))
```

```
dp = [1] * k
```

```
for i in range(1, k):  
    for j in range(i):  
        if missile_heights[i] <= missile_heights[j]:  
            dp[i] = max(dp[i], dp[j] + 1)
```

```
print(max(dp))
```

代码运行截图 <mark>（至少包含有"Accepted"）</mark>

状态: Accepted

源代码

```
k = int(input())
missile_heights = list(map(int, input().split()))

dp = [1] * k

for i in range(1, k):
    for j in range(i):
        if missile_heights[i] <= missile_heights[j]:
            dp[i] = max(dp[i], dp[j] + 1)

print(max(dp))
```

基本信息

#: 46894458
题目: 02945
提交人: EuphoriaJ
内存: 3584kB
时间: 26ms
语言: Python3
提交时间: 2024-11-01 22:29:39

©2002-2022 POJ 京ICP备20010980号-1

[English](#) [帮助](#) [关于](#)

23421: 小偷背包

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

思路：输入部分略，核心：创建一个二维列表 `dp` 并通过两层循环来填充 `dp`，对于每个 `i` 和 `j`，如果当前物品 `i-1` 的重量 `weight[i-1]` 大于背包容量 `j`，那么不能选择该物品，`dp[i][j]=dp[i-1][j]`；如果 `weight[i-1]<j`，那么就需要做出决策，即

`dp[i][j]=max(dp[i-1][j],dp[i - 1][j - weights[i - 1]] + prices[i - 1])`(2.5h)

代码:

```
N, B = map(int, input().split())  
prices = list(map(int, input().split()))  
weights = list(map(int, input().split()))
```

```
dp = [[0] * (B + 1) for _ in range(N + 1)]
```

```
for i in range(1, N + 1):  
    for j in range(1, B + 1):  
        if weights[i - 1] > j:  
            dp[i][j] = dp[i - 1][j]  
        else:  
            dp[i][j] = max(dp[i - 1][j], dp[i - 1][j -  
weights[i - 1]] + prices[i - 1])
```

```
print(dp[N][B])
```

代码运行截图 <mark>（至少包含有"Accepted"）</mark>

状态: Accepted

基

源代码

```
N, B = map(int, input().split())
prices = list(map(int, input().split()))
weights = list(map(int, input().split()))

dp = [[0] * (B + 1) for _ in range(N + 1)]

for i in range(1, N + 1):
    for j in range(1, B + 1):
        if weights[i - 1] > j:
            dp[i][j] = dp[i - 1][j]
        else:
            dp[i][j] = max(dp[i - 1][j], dp[i - 1][j - weights[i - 1]])

print(dp[N][B])
```

报

©2002-2022 POJ 京ICP备20010980号-1

02754: 八皇后

dfs and similar,

<http://cs101.openjudge.cn/practice/02754>

思路: 首先定义一个 `solve_n_queens` 函数来生成并返回所有 8 皇后问题的解。然后定义 `find_solution` 函数, 其接受一个索引 `b` 作为参数, 并返回第 `b` 个解。最后读取输入数据, 调用 `find_solution` 函数, 并打印结果即可。(4h)

代码:

```
def solve_n_queens(n):  
    def place_queen(row, columns, results):  
        if row == n:  
            results.append(''.join(str(col + 1) for col  
in columns))  
        else:  
            for col in range(n):  
                if all(abs(col - columns[i]) not in (0,  
row - i) for i in range(row)):  
                    place_queen(row + 1, columns + [col],  
results)  
            results = []  
            place_queen(0, [], results)  
            return sorted(results)  
solutions = solve_n_queens(8)  
def find_solution(b):  
    return solutions[b-1]  
n = int(input())  
for _ in range(n):
```

```
b = int(input())

print(find_solution(b))
```

代码运行截图 <mark>（至少包含有"Accepted"）</mark>

状态: Accepted

源代码

```
def solve_n_queens(n):
    def place_queen(row, columns, results):
        if row == n:
            results.append(''.join(str(col + 1) for col in columns))
        else:
            for col in range(n):
                if all(abs(col - columns[i]) not in (0, row - i) for i in range(row)):
                    place_queen(row + 1, columns + [col], results)
    results = []
    place_queen(0, [], results)
    return sorted(results)
solutions = solve_n_queens(8)
def find_solution(b):
    return solutions[b-1]
n = int(input())
for _ in range(n):
    b = int(input())
    print(find_solution(b))
```

基本信息

#: 46914915
题目: 02754
提交人: EuphoriaJ
内存: 3652kB
时间: 46ms
语言: Python3
提交时间: 2024-11-02 20:35:11

©2002-2022 POJ 京ICP备20010980号-1

[English](#) [帮助](#) [关于](#)

189A. Cut Ribbon

brute force, dp 1300

<https://codeforces.com/problemset/problem/189/A>

思路：核心：创建一个长度为 $n+1$ 的列表并将值初始为 -1 ($dp[0]$ 初始为 0)，再使用动态规划计算，当 $i \geq a$ 且 $dp[i-a] \neq -1$ 时， $dp[i] = \max(dp[i], dp[i-a] + 1)$ ；当 i 大于等于 b 且 $dp[i-b] \neq -1$ 时，与上述情况类似； c 也同理，最后输出 $dp[n]$ 即可(3.5h)

代码：

```
n, a, b, c = map(int, input().split())
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)
print(dp[n])
```

代码运行截图 <mark>（至少包含有"Accepted"）</mark>

Contest status							
#	When	Who	Problem	Lang	Verdict	Time	Memory
289443392	Nov/02/2024 20:31 UTC+8	EuphoriaJ	189A - Cut Ribbon	Python 3	Accepted	93 ms	0 KB

2. 学习总结和收获

<mark>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。</mark>

比上次更难了啊啊啊啊，没有 AI 帮助完全写不出来，况且将 AI 写的代码读懂就需要非常长的时间（上面写的耗时有绝大多数都是指读懂 AI 写的代码的时间），这周忙着复习线性代数，下周还要复习数学分析，只能先暂时把记概搁一搁，等到期中后再把每日选做补回来（有点慌，感觉期末要挂了 qwq）

