

俄罗斯套娃信封问题

给你一个二维整数数组 `envelopes`，其中 `envelopes[i] = [wi, hi]`，表示第 *i* 个信封的宽度和高度。

当另一个信封的宽度和高度都比这个信封大的时候，这个信封就可以放进另一个信封里，如同俄罗斯套娃一样。

请计算 **最多能有多少个** 信封能组成一组“俄罗斯套娃”信封（即可以把一个信封放到另一个信封里面）。

注意：不允许旋转信封。

示例 1：

输入：`envelopes = [[5,4],[6,4],[6,7],[2,3]]`

输出：3

解释：最多信封的个数为 3，组合为：`[2,3] => [5,4] => [6,7]`。

示例 2：

输入：`envelopes = [[1,1],[1,1],[1,1]]`

输出：1

```
class Solution:
    def maxEnvelopes(self, envelopes: List[List[int]]) -> int:
        #按照保证一维升序基础上 一维相同情况下进行二维降序
        #按照排序序列利用动态规划进行连续最长递增子序列查找
        if not envelopes:
            return 0
        N=len(envelopes)
        envelopes.sort(key=lambda x:(x[0],-x[1]))#lambda 表达式
        dp=[1]*N
        for i in range(N):
            for j in range(i):#定义 dp[i] 表示以 i 结尾的最长递增子序列的长度。
                对每个 i 的位置，遍历 [0, i)
                if envelopes[j][1]<envelopes[i][1]:
                    dp[i]=max(dp[i],dp[j]+1)#只考虑第二维度递增 通过 dp 计数+1
        return max(dp)
```