Solution 1

```python
# Copyright © 2020 AlgoExpert, LLC. All rights reserved.

# O(n^2) time | O(n) space
def diskStacking(disks):
    disks.sort(key=lambda disk: disk[2])
    heights = [disk[2] for disk in disks]
    sequences = [None for disk in disks]
    maxHeightIdx = 0
    for i in range(1, len(disks)):
        currentDisk = disks[i]
        for j in range(0, i):
            otherDisk = disks[j]
            if areValidDimensions(otherDisk, currentDisk):
                if heights[i] <= currentDisk[2] + heights[j]:
                    heights[i] = currentDisk[2] + heights[j]
                    sequences[i] = j
        if heights[i] >= heights[maxHeightIdx]:
            maxHeightIdx = i
    return buildSequence(disks, sequences, maxHeightIdx)


def areValidDimensions(o, c):
    return o[0] < c[0] and o[1] < c[1] and o[2] < c[2]


def buildSequence(array, sequences, currentIdx):
    sequence = []
    while currentIdx is not None:
        sequence.append(array[currentIdx])
        currentIdx = sequences[currentIdx]
    return list(reversed(sequence))

```