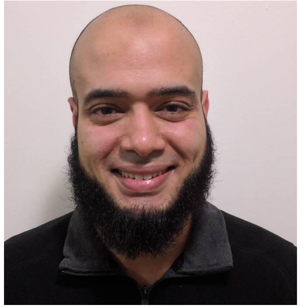
Algorithms

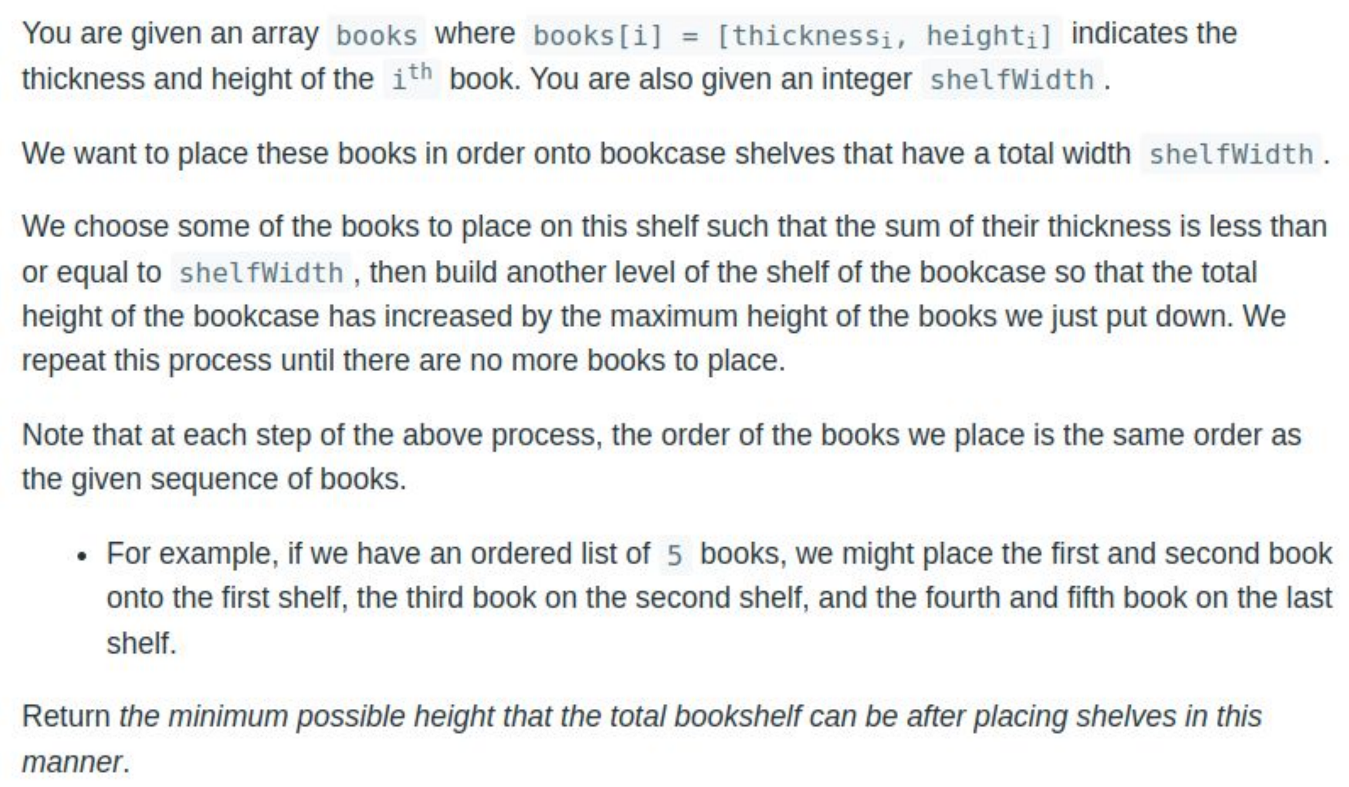
DP Homework 1

Mostafa S. Ibrahim
Teaching, Training and Coaching for more than a decade!

Artificial Intelligence & Computer Vision Researcher
PhD from Simon Fraser University - Canada
Bachelor / Msc from Cairo University - Egypt
Ex-(Software Engineer / ICPC World Finalist)



Problem #1: LeetCode 1105 - Filling Bookcase Shelves



You are given an array `books` where `books[i] = [thickness_i, height_i]` indicates the thickness and height of the i^{th} book. You are also given an integer `shelfWidth`.

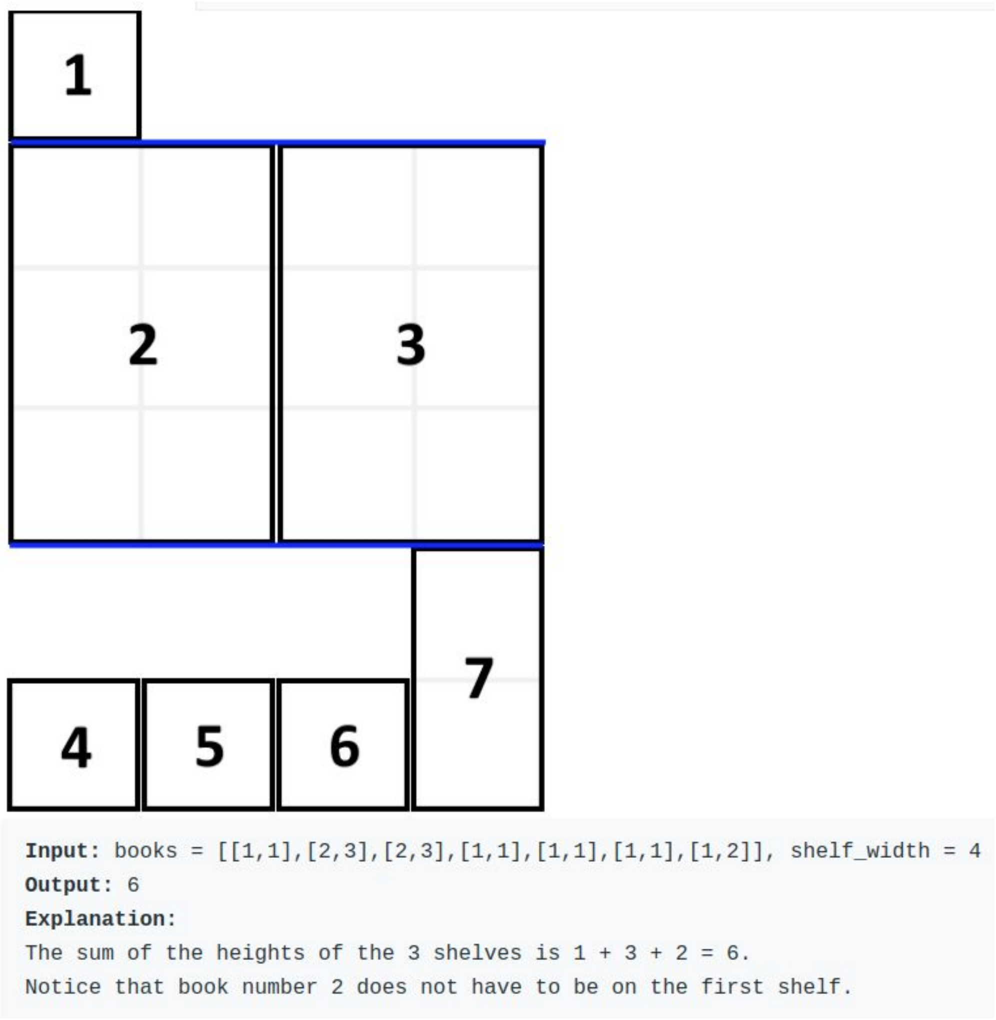
We want to place these books in order onto bookcase shelves that have a total width `shelfWidth`.

We choose some of the books to place on this shelf such that the sum of their thickness is less than or equal to `shelfWidth`, then build another level of the shelf of the bookcase so that the total height of the bookcase has increased by the maximum height of the books we just put down. We repeat this process until there are no more books to place.

Note that at each step of the above process, the order of the books we place is the same order as the given sequence of books.

- For example, if we have an ordered list of `5` books, we might place the first and second book onto the first shelf, the third book on the second shelf, and the fourth and fifth book on the last shelf.

Return *the minimum possible height that the total bookshelf can be after placing shelves in this manner*.



```
Input: books = [[1,1],[2,3],[2,3],[1,1],[1,1],[1,1],[1,2]], shelf_width = 4
Output: 6
Explanation:
The sum of the heights of the 3 shelves is 1 + 3 + 2 = 6.
Notice that book number 2 does not have to be on the first shelf.
```

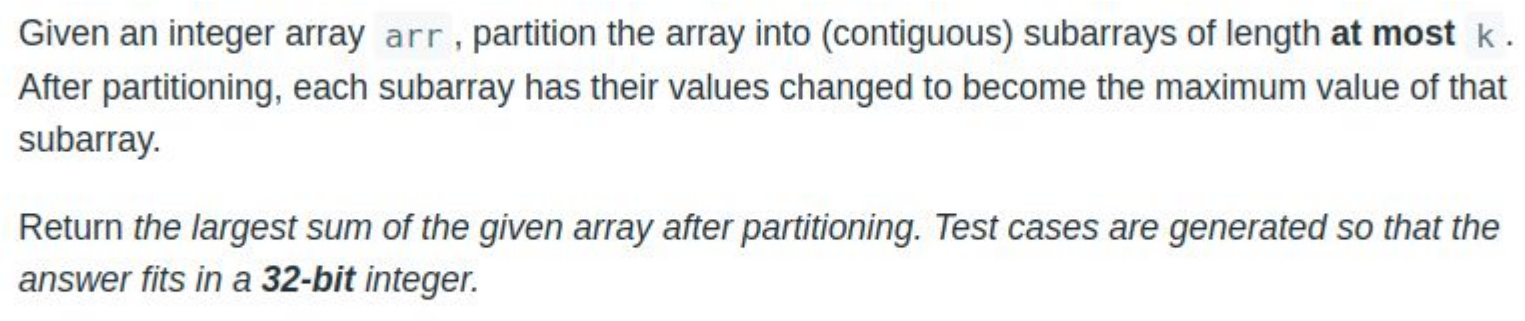
Example 2:

```
Input: books = [[1,3],[2,4],[3,2]], shelfWidth = 6
Output: 4
```

Constraints:

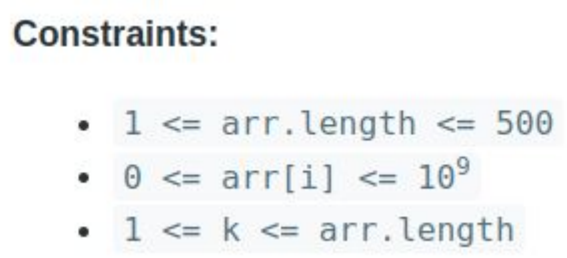
- `1 <= books.length <= 1000`
- 1 <= thickness$_i$ <= shelfWidth <= 1000
- 1 <= height$_i$ <= 1000

Problem #2: LeetCode 1043 - Partition Array for Maximum Sum



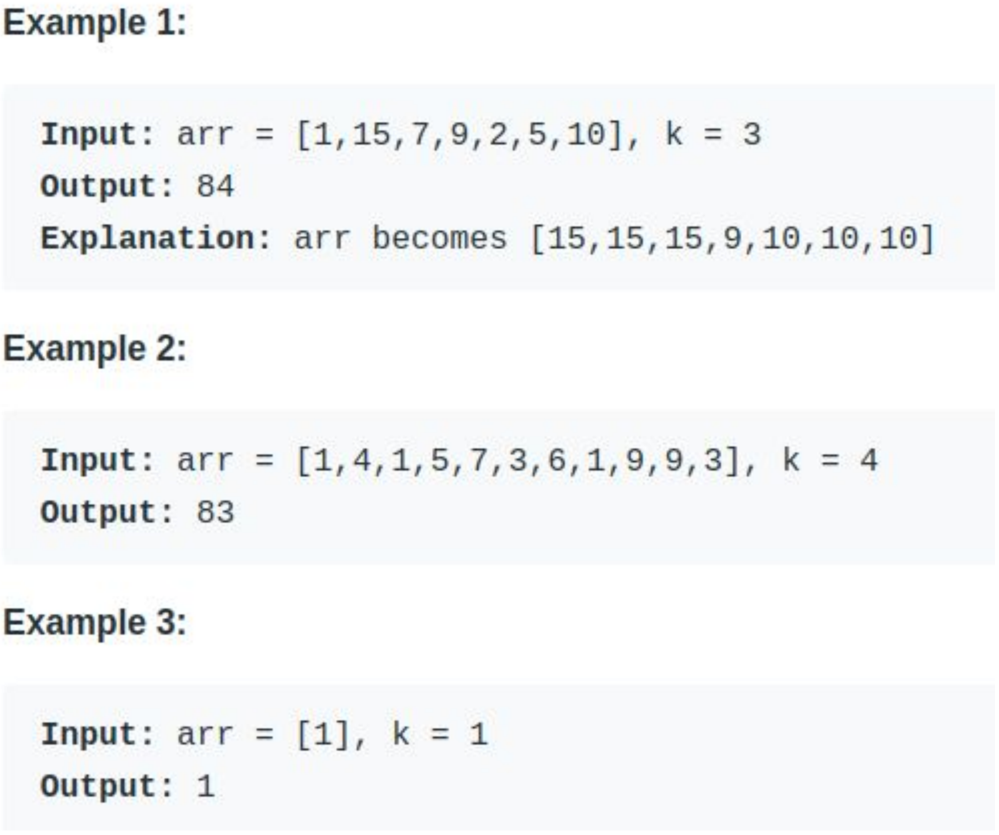
Given an integer array `arr`, partition the array into (contiguous) subarrays of length **at most** `k`. After partitioning, each subarray has their values changed to become the maximum value of that subarray.

Return *the largest sum of the given array after partitioning. Test cases are generated so that the answer fits in a* ***32-bit*** *integer.*



Constraints:

- `1 <= arr.length <= 500`
- `0 <= arr[i] <=` 10^9
- `1 <= k <= arr.length`



Example 1:

```
Input: arr = [1,15,7,9,2,5,10], k = 3
Output: 84
Explanation: arr becomes [15,15,15,9,10,10,10]
```

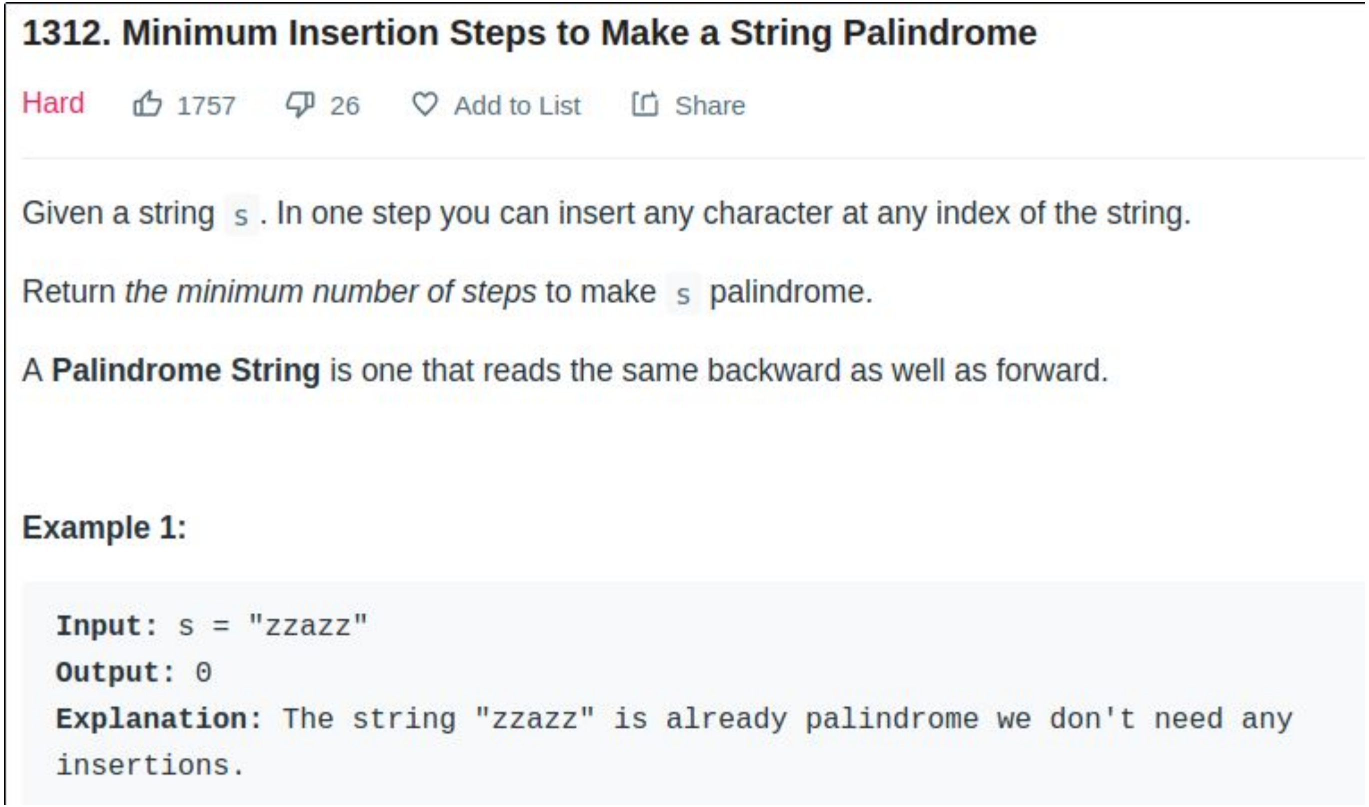
Example 2:

```
Input: arr = [1,4,1,5,7,3,6,1,9,9,3], k = 4
Output: 83
```

Example 3:

```
Input: arr = [1], k = 1
Output: 1
```

Problem #3: LeetCode 1312 - Minimum Insertion Steps to Make a String Palindrome



1312. Minimum Insertion Steps to Make a String Palindrome

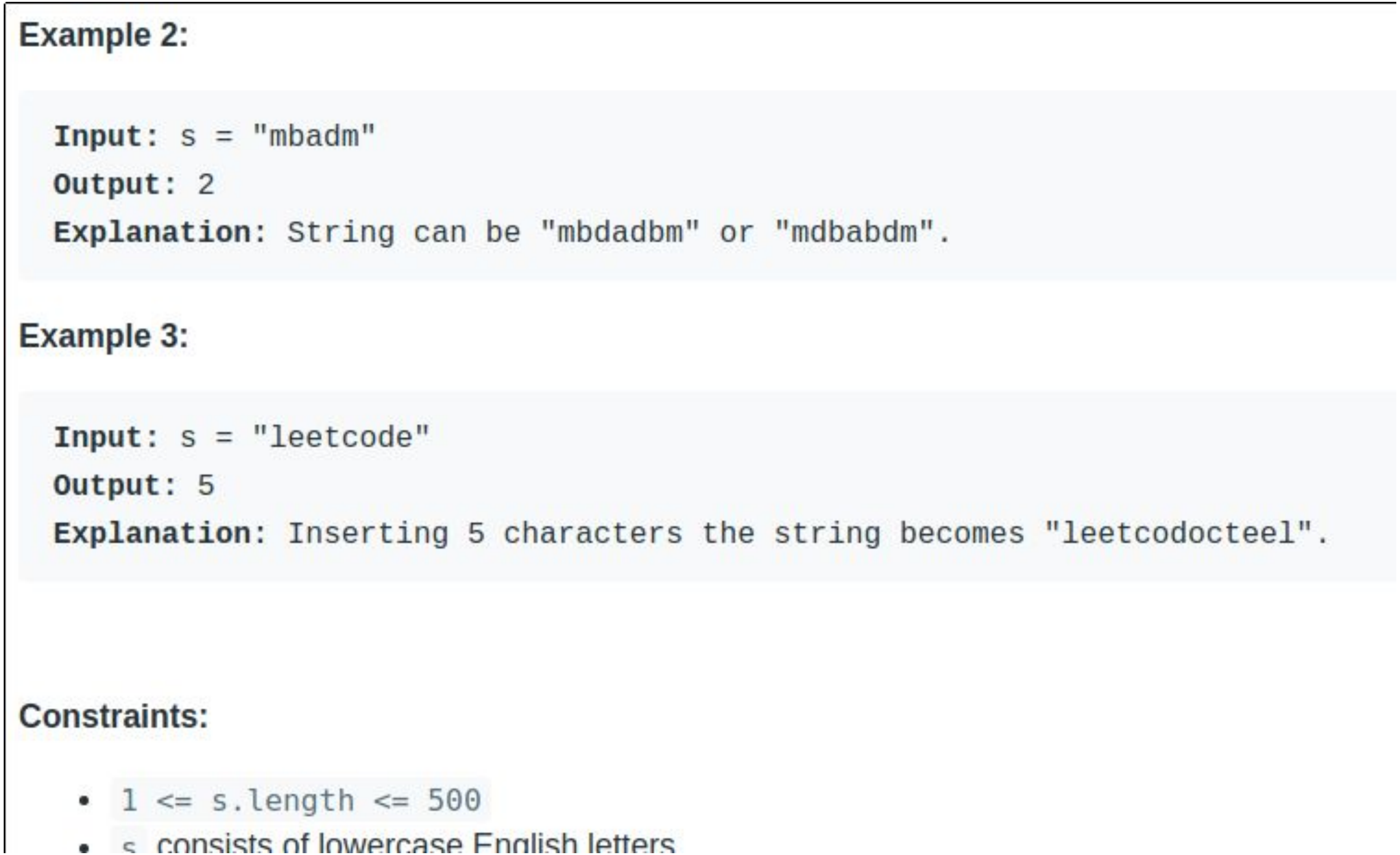
Given a string `s`. In one step you can insert any character at any index of the string.

Return *the minimum number of steps* to make `s` palindrome.

A **Palindrome String** is one that reads the same backward as well as forward.

Example 1:

```
Input: s = "zzazz"
Output: 0
Explanation: The string "zzazz" is already palindrome we don't need any
insertions.
```



Example 2:

```
Input: s = "mbadm"
Output: 2
Explanation: String can be "mbdadbm" or "mdbabdm".
```

Example 3:

```
Input: s = "leetcode"
Output: 5
Explanation: Inserting 5 characters the string becomes "leetcodocteel".
```

Constraints:

- `1 <= s.length <= 500`
- `s` consists of lowercase English letters

"Acquire knowledge and impart it to the people."

"Seek knowledge from the Cradle to the Grave."