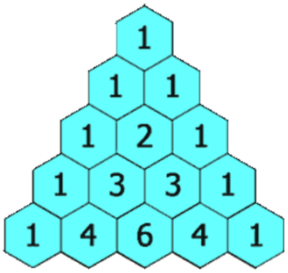
2023-09-09 - Handout – Dynamic Programming

Q1. Pascal's Triangle II

Link: https://leetcode.com/problems/pascals-triangle-ii/



Given an integer `rowIndex`, return the `rowIndex`th (**0-indexed**) row of the **Pascal's triangle**.

In **Pascal's triangle**, each number is the sum of the two numbers directly above it as shown:

```
Input: rowIndex = 3
Output: [1,3,3,1]
```

```
Input: rowIndex = 0
Output: [1]
```

Q2. Word Break

Link: https://leetcode.com/problems/word-break/

Given a string `s` and a dictionary of strings `wordDict`, return `true` if `s` can be segmented into a space-separated sequence of one or more dictionary words.

Note that the same word in the dictionary may be reused multiple times in the segmentation.

```
Input: s = "catsandog", wordDict = ["cats","dog","sand","and","cat"]
Output: false
```

Constraints:

- `1 <= s.length <= 300`
- `1 <= wordDict.length <= 1000`
- `1 <= wordDict[i].length <= 20`
- `s` and `wordDict[i]` consist of only lowercase English letters.
- All the strings of `wordDict` are **unique**.

Q3. Sort Integers by the Power Value

Link: https://leetcode.com/problems/sort-integers-by-the-power-value/description/

The power of an integer `x` is defined as the number of steps needed to transform `x` into `1` using the following steps:

- if `x` is even then `x = x / 2`
- if `x` is odd then `x = 3 * x + 1`

For example, the power of `x = 3` is `7` because `3` needs `7` steps to become `1` (`3 --> 10 --> 5 --> 16 --> 8 --> 4 --> 2 --> 1`).

Given three integers `lo`, `hi` and `k`. The task is to sort all integers in the interval `[lo, hi]` by the power value in **ascending order**, if two or more integers have **the same** power value sort them by **ascending order**.

Return the `k`th integer in the range `[lo, hi]` sorted by the power value.

Notice that for any integer `x` `(lo <= x <= hi)` it is **guaranteed** that `x` will transform into `1` using these steps and that the power of `x` is will **fit** in a 32-bit signed integer.

```
Input: lo = 12, hi = 15, k = 2
Output: 13
```

```
Input: lo = 7, hi = 11, k = 4
Output: 7
```

Constraints:

- `1 <= lo <= hi <= 1000`
- `1 <= k <= hi - lo + 1`

Q4. Maximum Value of K coins From Piles

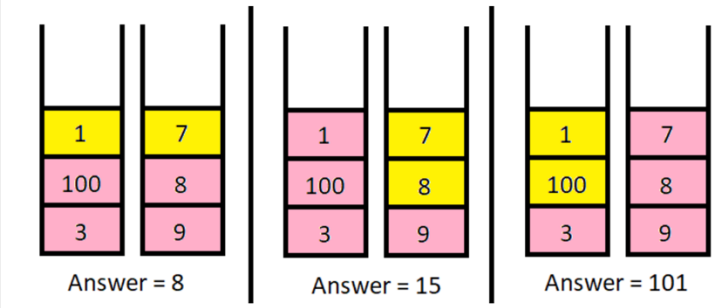
Link: https://leetcode.com/problems/maximum-value-of-k-coins-from-piles/

There are `n` **piles** of coins on a table. Each pile consists of a **positive number** of coins of assorted denominations.

In one move, you can choose any coin on **top** of any pile, remove it, and add it to your wallet.

Given a list `piles`, where `piles[i]` is a list of integers denoting the composition of the `i`th pile from **top to bottom**, and a positive integer `k`, return *the **maximum total value** of coins you can have in your wallet if you choose **exactly** `k` coins optimally*.

Example 1:




```
Input: piles = [[1,100,3],[7,8,9]], k = 2
Output: 101
```

Constraints:

- `n == piles.length`
- `1 <= n <= 1000`
- `1 <= piles[i][j] <= 10`^{5}
- `1 <= k <= sum(piles[i].length) <= 2000`