Problem # 948 : Bag of Tokens

<https://leetcode.com/problems/bag-of-tokens/>

Solution:

1. Let n be the number of elements in tokens.

2. If n is less than 1 return 0.

3. If n is equal to 1 and if the token is more than power P, return 0.

Otherwise return 1.

4. Sort tokens array in ascending order.

Initialize the left pointer i to 0, the right pointer j to n – 1 (i.e. the last element in tokens array).

Initialize score and max\_score to 0.

5. Iterate using a while loop for the condition that I is less than or equal to j.

6. If power P is more than or equal to the token value at index I, increment the score by 1,

decrease P by the value of the token, increment i pointer and max\_score will be the maximum of max\_score and score.

7. If P is less than token value at index and if score is positive, then increment P by the value of the token, and decrement the score by 1. Decrement the j pointer.

8. If this is not the case, then break from the while loop.

9. Return max\_score.

class Solution:

def bagOfTokensScore(self, tokens: List[int], P: int) -> int:

n = len(tokens)

if n < 1:

return 0

elif n == 1:

if tokens[0] > P:

return 0

else:

return 1

tokens = sorted(tokens) # sort in ascending order

i = 0

j = n-1

score = 0

max\_score = 0

while (i <= j):

if P >= tokens[i]: # play face up, losing tokens[i] power and gaining 1 score

score += 1

max\_score = max(max\_score, score)

P -= tokens[i]

i += 1

elif score: # play face down, gaining tokens[j] power and losing 1 score

P += tokens[j]

score -= 1

j -= 1

else:

break

return(max\_score)