Input: nums : int[]
k : int

Output: Count of partitions s.t. $|\max(part) - \min(part)| \leq k$

---

Bruteforce

count partitions incrementally for [0,1), [0,2), ... [0,n)

Algorithm:

dp[i] is "count of partition [0, i)"

for each i : 1 → n
  for each j : i → 0  # go backward
    dp[i] += dp[j-1]  if valid

ret dp[n]  ⇒ $O(n^2)$

---

Optimized

Insight 1: Assume all partitions are valid, the distribution of answer space is:

| [0,1) | [0,2) | [0,3) | [0,4) | [0,5) | ... | [0,n) |
|---|---|---|---|---|---|---|
| 1 | 2 | 4 | 8 | ... | | $2^{n-1}$ |

↳ prefix sum

So if we assume all valid, then complexity becomes O(n) with prefix sum

Insight 2: Now if we have an efficient way to check for condition, then we DONE

"what is efficient way to get min/max of a partition?"

⇒ heap