Problem # 495: Teemo Attacking

My solution:

https://leetcode.com/problems/teemo-attacking/discuss/865713/Simple-Python-3-Solution-Runtime-beats-98.57

1. If not timeSeries (i.e. timeSeries list is empty) return 0.
2. If timeSeries has only one element, return duration.
3. For more than 1 element in the timeSeries, initialize count to 0.
4. Take the difference between each element and its preceeding element. If this difference is less than the duration, then take this difference and add it to count. However, if the difference is greater than or equal to the duration, then add the duration to count.
5. For the last element in the timeSeries, add duration to count.
6. Return count.

class Solution:

def findPoisonedDuration(self, timeSeries: List[int], duration: int) -> int:

if not timeSeries:

return(0)

if len(timeSeries) == 1:

return(duration)

count = 0

for i in range(1, len(timeSeries)):

count += min(duration, timeSeries[i] - timeSeries[i-1])

count += duration # for the last item in the timeSeries

return(count)

Runtime for the above code beats 100% in runtime.

1. Please see given below shorter code using list comprehension:

class Solution:

def findPoisonedDuration(self, timeSeries: List[int], duration: int) -> int:

if not timeSeries:

return(0)

if len(timeSeries) == 1:

return(duration)

return(duration + sum([min(duration, timeSeries[i] - timeSeries[i-1]) for i in range(1, len(timeSeries))]))

Runtime for the list comprehension only beats 98.57%