Problem #2515: Shortest Distance to Target String in a Circular Array

<https://leetcode.com/problems/shortest-distance-to-target-string-in-a-circular-array/>

My Solution:

1. If target not in words, return -1.
2. If the word at startIndiex is the same as target, then return 0.
3. For circular array effect, prepend and append words to words, i.e. Make newWords by concatenating words 3 times.
4. Now startIndex is shifted by length of words.
5. Set left and right pointers to be at startIndex.
6. Initialize leftDist and rightDist to 0.
7. while left pointer is at 0 or greater and newWords at left pointer index is not equal to target, move left by decrementing by 1 and increment leftDist by 1.
8. while right pointer less than the last index and newWords at right pointer index is not equal to target, move right by incrementing by 1 and increment leftDist by 1.
9. Return the minimum of leftDist and rightDist.

class Solution:

def closetTarget(self, words: List[str], target: str, startIndex: int) -> int:

if target not in words:

return -1

if words[startIndex] == target:

return 0

newWords = words \* 3

startIndex = len(words) + startIndex

left = startIndex

right = startIndex

leftDist = 0

rightDist = 0

while left >= 0 and newWords[left] != target:

left -= 1

leftDist += 1

while right <= len(newWords) - 1 and newWords[right] != target:

right += 1

rightDist += 1

return min(leftDist, rightDist)