**Problem #1629: Slowest Key**

-----------------------------------------

<https://leetcode.com/problems/slowest-key/>

My Solution:

1. Initialize max\_time to releaseTimes at index 0 and max\_key to keysPressed at index 0.

2. Let n be the length of releaseTimes list.

3. Iterate through the list from index 1 through n and do the following:

1. Calculate time which is the releaseTimes from index i-1 to index i.
2. If time is greater than max\_time, update max\_key to the present KeysPressed and max\_time to time.
3. If time is equal to max\_time and if the present KeysPressed is greater than max\_key
4. (i.e., lexicographically greater) then update max\_key to the present KeysPressed and max\_time to time.

4. Finally return max\_key.

------------------------------------------------------

class Solution:

def slowestKey(self, releaseTimes: List[int], keysPressed: str) -> str:

max\_time = releaseTimes[0]

max\_key = keysPressed[0]

n = len(releaseTimes)

for i in range(1, n):

time = releaseTimes[i] - releaseTimes[i-1]

if time > max\_time or (time == max\_time and keysPressed[i] > max\_key):

max\_key = keysPressed[i]

max\_time = time

return max\_key

--------------------------------------------------------------------------------------------------------------