Two Sum/HashMap

Question: https://leetcode.com/problems/two-sum/

Given an array of integers `nums` and an integer `target`, return indices of the two numbers such that they add up to `target`.

You may assume that each input would have exactly one solution, and you may not use the same element twice.

You can return the answer in any order.

Example 1:

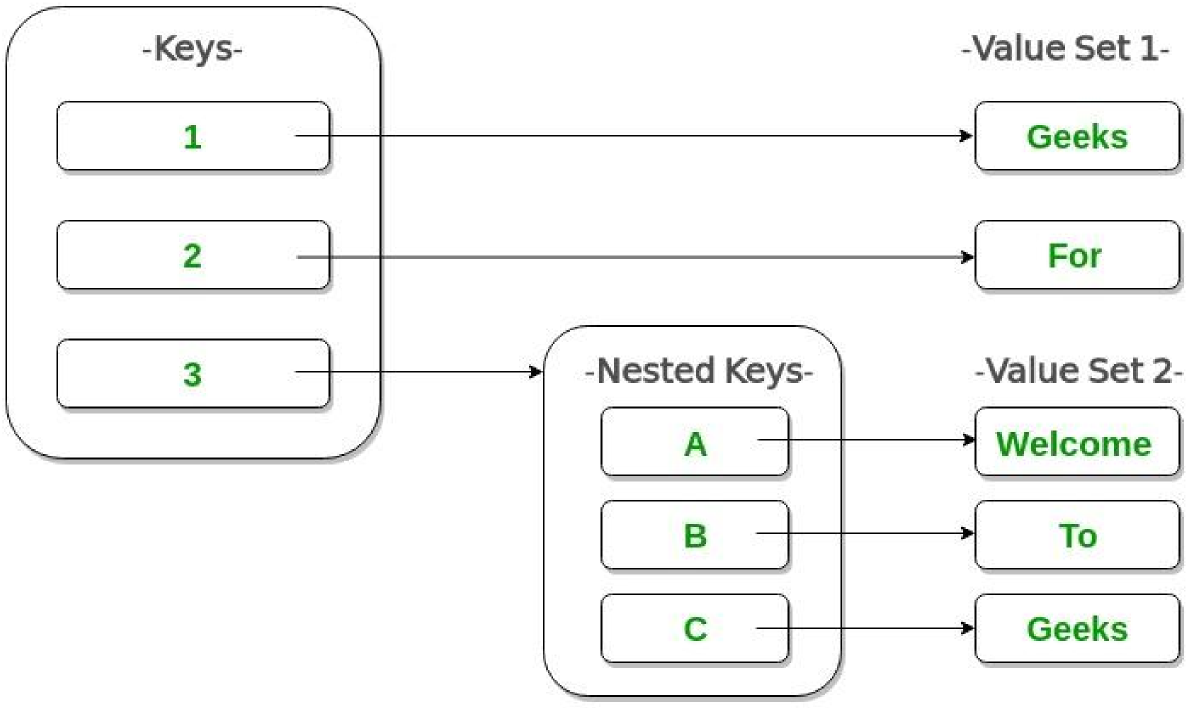
```
Input: nums = [2,7,11,15], target = 9
Output: [0,1]
Explanation: Because nums[0] + nums[1] == 9, we return [0, 1].
```

Since we need 2 pieces of information about a number

1. The number itself
2. Another number which after addition return the target Sum

Therefore we will use a hashmap for this question.
In Python, dictionaries are implemented as hashmaps in the background which allows us to directly use a dictionary as a hashmap with Key: Value pairs.

To read more on dictionaries as hashmaps:
https://www.geeksforgeeks.org/hash-map-in-python/



PS: Whenever you need 2 pieces of information for something such as Employee Name and Employee ID use hashmaps.

The best part about hashtables or hashmaps is that data access takes **O(1)** time hence searching takes **O(1)** time which is really beneficial for this problem.

So my approach was to loop over the array nums and to store 2 pieces of information:
the number itself and
Target-number which I stored in the adder variable
Using hashmap, I store it as key: value where the key is the number and the value is the adder

For every number, I check if the adder is already present as a key, if yes then I return the number index and adder index,
Else I store the number: index

<u>MY SOLUTION:</u>

```
b={}
for i in range(len(nums)):
        adder = target - nums[i]
```

```
if adder in b:
        return [b[adder],i]
else:
        b[nums[i]] = i
```

Code : https://gist.github.com/vermaayush680/4956b817287849754fdf9758a9e0127a

The best part about hashmaps is that the line
If adder in b: takes O(1) time to check because searching takes O(1) time in hashmaps.

Comment any suggestions you have for me or better approaches you have for this question.