Problem # 20: Valid Parentheses. (Easy)

<https://leetcode.com/problems/valid-parentheses/>

My Solution:

1. Initialize a set called left\_bracs with "(", "{", and "[", and another set called right\_bracs with ")", "}", and "]".

2. Initialize an empty list called a stack.

3. Iterate through the string character by character.

4. If the character is in left\_bracs append it to the stack. Otherwise, the character is in right\_bracs. If stack is empty, then return False.

If stack is not empty, pop the stack. If the stack does not have the corresponding left parenthesis type, then return False.

5. Finally after iterating through the array, the stack must be empty if the parentheses in the left are balanced with those in the right.

If stack is empty, return True. Return False otherwise.

class Solution:

def isValid(self, s: str) -> bool:

left\_bracs = set(["(", "{", "["])

right\_bracs = set([")", "}", "]"])

stack = []

n = len(s)

for i in range(n):

if s[i] in left\_bracs:

stack.append(s[i])

else: # s[i] is a right bracket

if not stack:

return(False)

else: # stack is not empty

if (s[i] == ']' and stack.pop() != '[') or \

(s[i] == ')' and stack.pop() != '(') or \

(s[i] == '}' and stack.pop() != '{'):

return(False)

return(not stack)