Construct the Rectangle - LeetCode
leetcode.com

Memory usage: 80.3 MB

leetcode.com/problems/construct-the-

Problem List

Submit

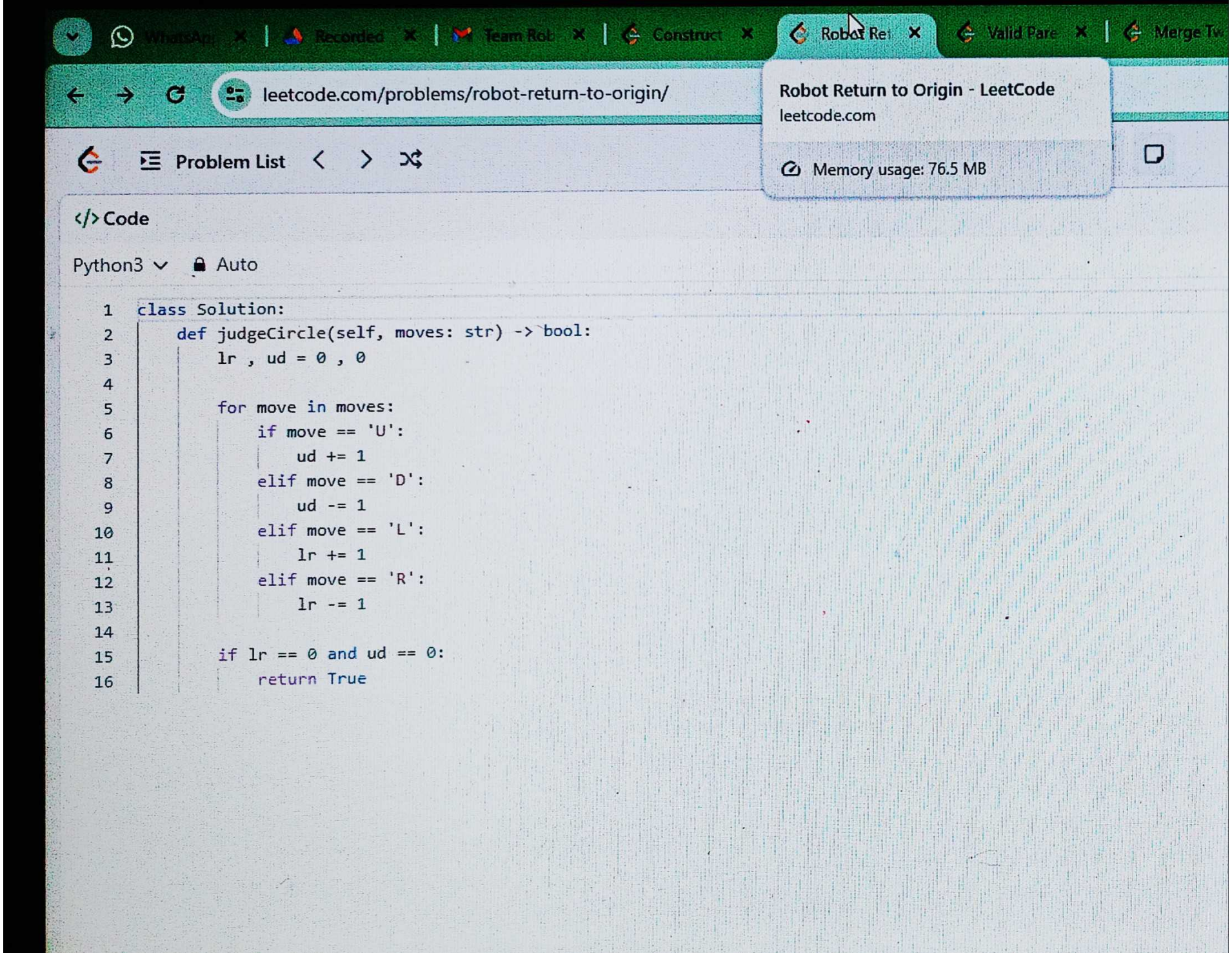
Code

Python3 Auto

```python
class Solution:
    def constructRectangle(self, area: int) -> List[int]:
        # initialize width and length with square root of target area rounded down and up, respectively
        width = int(math.sqrt(area))
        length = int(math.ceil(area / width))

        # loop until width and length multiplied is equal to target area
        while width * length != area:
            # if width * length is less than target area, increment length
            if width * length < area:
                length += 1
            # if width * length is greater than target area, decrement width
            else:
                width -= 1

        # return [length, width]
        return [length, width]
```


leetcode.com/problems/robot-return-to-origin/
Robot Return to Origin - LeetCode
leetcode.com
Memory usage: 76.5 MB
Problem List
Code
Python3
Auto
class Solution:
def judgeCircle(self, moves: str) -> bool:
lr , ud = 0 , 0
for move in moves:
if move == 'U':
ud += 1
elif move == 'D':
ud -= 1
elif move == 'L':
lr += 1
elif move == 'R':
lr -= 1
if lr == 0 and ud == 0:
return True

leetcode.com/problems/pascals-triangle/

Pascal's Triangle - LeetCode
leetcode.com

Memory usage: 221 MB

Problem List

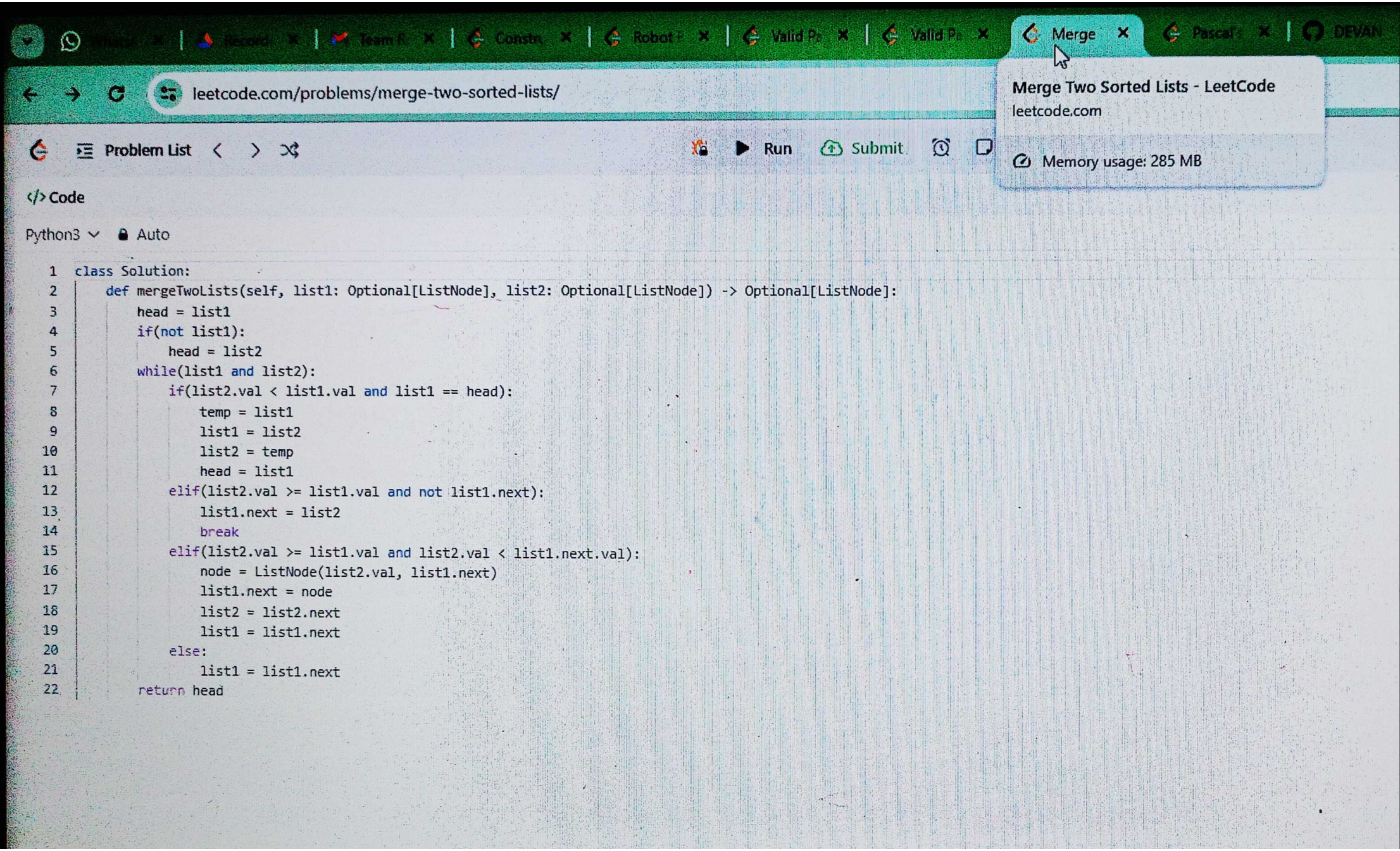
Run Submit

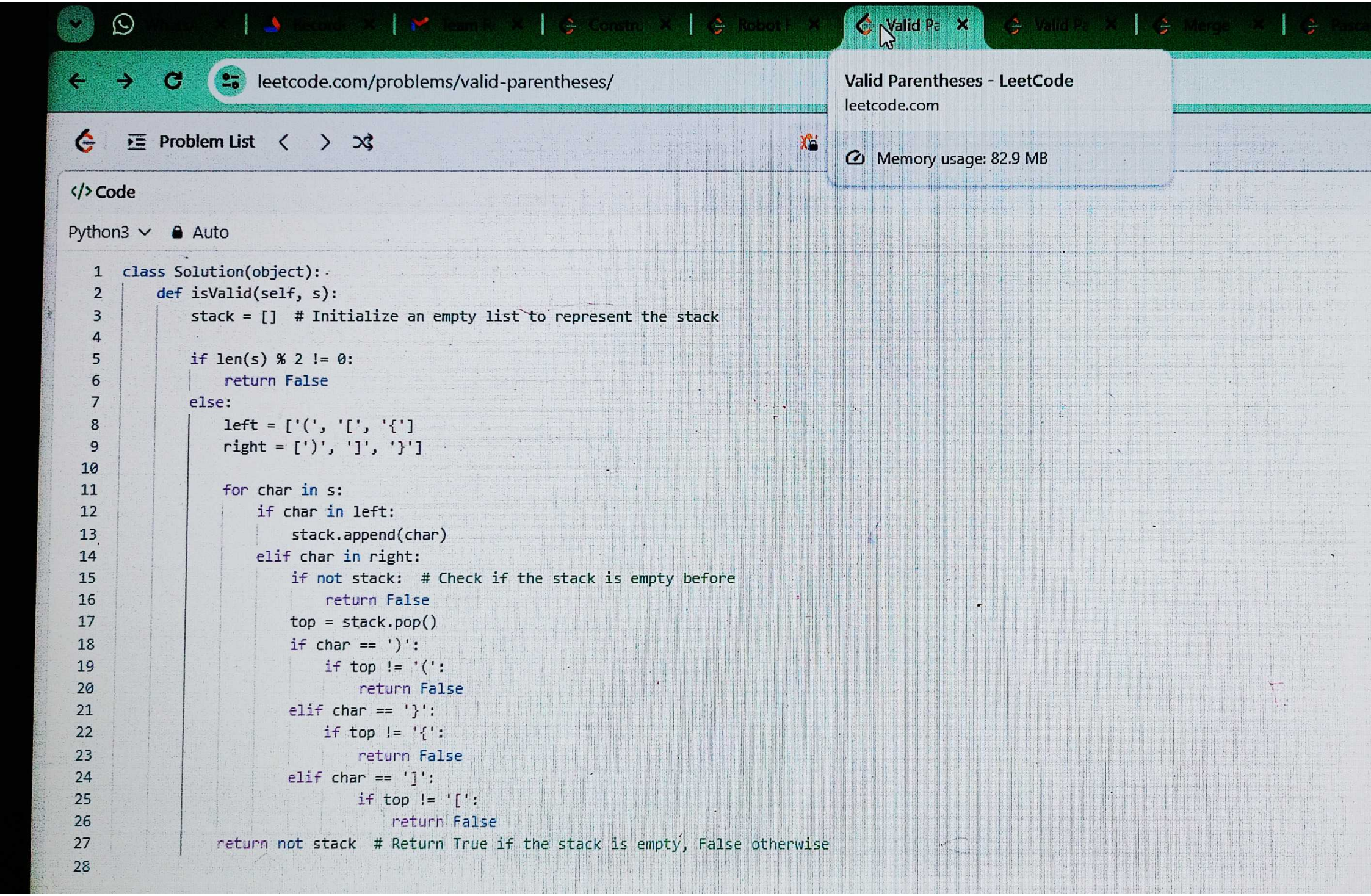
Code

Python3 Auto

```python
from typing import List

class Solution:
    def generate(self, numRows: int) -> List[List[int]]:
        finalNums = []
        finalNums.append([1])
        for i in range(numRows - 1):
            newRow = [1]
            for j in range(i):
                newRow.append(finalNums[i][j] + finalNums[i][j + 1])
            newRow.append(1)
            finalNums.append(newRow)
        return finalNums
```


Merge
Pascal
DEVAN
leetcode.com/problems/merge-two-sorted-lists/
Merge Two Sorted Lists - LeetCode
leetcode.com
Memory usage: 285 MB
Problem List
Run
Submit
Code
Python3
Auto
class Solution:
def mergeTwoLists(self, list1: Optional[ListNode], list2: Optional[ListNode]) -> Optional[ListNode]:
head = list1
if(not list1):
head = list2
while(list1 and list2):
if(list2.val < list1.val and list1 == head):
temp = list1
list1 = list2
list2 = temp
head = list1
elif(list2.val >= list1.val and not list1.next):
list1.next = list2
break
elif(list2.val >= list1.val and list2.val < list1.next.val):
node = ListNode(list2.val, list1.next)
list1.next = node
list2 = list2.next
list1 = list1.next
else:
list1 = list1.next
return head


Valid Pa
leetcode.com/problems/valid-parentheses/
Valid Parentheses - LeetCode
leetcode.com
Memory usage: 82.9 MB
Problem List
Code
Python3
Auto
class Solution(object):
def isValid(self, s):
stack = [] # Initialize an empty list to represent the stack
if len(s) % 2 != 0:
return False
else:
left = ['(', '[', '{']
right = [')', ']', '}']
for char in s:
if char in left:
stack.append(char)
elif char in right:
if not stack: # Check if the stack is empty before
return False
top = stack.pop()
if char == ')':
if top != '(':
return False
elif char == '}':
if top != '{':
return False
elif char == ']':
if top != '[':
return False
return not stack # Return True if the stack is empty, False otherwise