Q1

class Solution:

def construct2DArray(self, original: List[int], m: int, n: int) -> List[List[int]]:

if len(original)!=m\*n:

return []

matrix=[]

for i in range(0,len(original),n):

matrix.append(original[i:i+n])

return matrix

Q2

class Solution:

def arrangeCoins(self, n: int) -> int:

left, right = 1, n

while left <= right:

mid = left + (right - left) // 2

coins = (mid \* (mid + 1)) // 2

if coins <= n:

left = mid + 1

else:

right = mid - 1

return right

Q3

class Solution:

def sortedSquares(self, nums: List[int]) -> List[int]:

return sorted([num\*num for num in nums])

Q4

class Solution:

def findDifference(self, nums1: List[int], nums2: List[int]) -> List[List[int]]:

num1,num2 = set(nums1),set(nums2)

res1,res2 = set(), set()

for n in nums1:

if n not in num2:

res1.add(n)

for n in nums2:

if n not in num1:

res2.add(n)

return [list(res1),list(res2)]

Q5

class Solution:

def findTheDistanceValue(self, arr1: List[int], arr2: List[int], d: int) -> int:

arr2.sort()

c =0

for i in range(len(arr1)):

l = 0; r = len(arr2)-1

while l<=r:

mid = l + (r-l) //2

if abs(arr1[i] - arr2[mid]) <= d:

break

elif arr2[mid] < arr1[i]:

l = mid + 1

else:

r = mid - 1

c += l>r

return c

Q6

class Solution:

def findDuplicates(self, nums: List[int]) -> List[int]:

output = []

for n in nums:

m = abs(n)

if nums[m-1] < 0:

output.append(m)

else:

nums[m-1] \*= -1

return output

Q7

class Solution:

def findMin(self, nums: List[int]) -> int:

hi, lo = len(nums) - 1, 0

while hi - 1 > lo:

mid = (hi + lo)//2

if nums[lo] > nums[mid]:

hi = mid

else:

lo = mid

if nums[hi] > nums[lo]:

return nums[0]

return nums[hi]

Q8

class Solution:

def findOriginalArray(self, changed: List[int]) -> List[int]:

changed.sort()

stk,res=deque([]),[]

for i in changed:

if stk and stk[0]\*2==i:

b=stk.popleft()

res.append(b)

else:

stk.append(i)

return res if not stk else []