Q1. Given three integer arrays arr1, arr2 and arr3 sorted in strictly increasing order, return a sorted array of only the integers that appeared in all three arrays.  
  
def Common(ar1, ar2, ar3, n1, n2, n3):

lst=[]

i, j, k = 0, 0, 0

while (i < n1 and j < n2 and k < n3):

if (ar1[i] == ar2[j] and ar2[j] == ar3[k]):

lst.add(ar1[i])

i += 1

j += 1

k += 1

elif ar1[i] < ar2[j]:

i += 1

elif ar2[j] < ar3[k]:

j += 1

else:

k += 1

return lst

Q2 Given two \*\*0-indexed\*\* integer arrays nums1 and nums2, return \*a list\* answer \*of size\* 2 \*where:\* - answer[0] \*is a list of all \*\*distinct\*\* integers in\* nums1 \*which are \*\*not\*\* present in\* nums2\*.\*answer[1] \*is a list of all \*\*distinct\*\* integers in\* nums2 \*which are \*\*not\*\* present in\* nums1.Note\*\* that the integers in the lists may be returned in \*\*any\*\* order.

class Solution:

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

n1=set(nums1)

n2=set(nums2)

r1=list(set(x for x in nums1 if x not in n2))

r2=list(set(x for x in nums2 if x not in n1))

return [r1,r2]

**Question 3** Given a 2D integer array matrix, return the ***transpose*** of matrix.

The **transpose** of a matrix is the matrix flipped over its main diagonal, switching the matrix's row and column indices

Answer.

class Solution:

def transpose(self, matrix: List[List[int]]) -> List[List[int]]:

m,n=len(matrix),len(matrix[0])

ans = [[None] \* m for \_ in range(n)]

for i in range(m):

for j in range(n):

ans[j][i]=matrix[i][j]

return ans

Q4. Given an integer array nums of 2n integers, group these integers into n pairs (a1, b1), (a2, b2), ..., (an, bn) such that the sum of min(ai, bi) for all i is **maximized**. Return *the maximized sum*.

def arrayPairSum(self, nums):

nums.sort()

result = 0

Legth = len(nums)

for i in range(0, Legth - 1, 2):

result += nums[i]

return result

Q5. You have n coins and you want to build a staircase with these coins. The staircase consists of k rows where the ith row has exactly i coins. The last row of the staircase \*\*may be\*\* incomplete.Given the integer n, return \*the number of \*\*complete rows\*\* of the staircase you will build\*.

class Solution:

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

return int(sqrt(2 \* n + 0.25) - 0.50)

Q6. Given an integer array nums sorted in non-decreasing order, return an array of the squares of each number sorted in non-decreasing order.

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

return\_array = [v\*\*2 for v in A]

return\_array.sort()

return return\_array

Q7. You are given an m x n matrix M initialized with all 0's and an array of operations ops, where ops[i] = [ai, bi] means M[x][y] should be incremented by one for all 0 <= x < ai and 0 <= y < bi.Count and return the number of maximum integers in the matrix after performing all the operations

Answer.

|  |  |
| --- | --- |
|  | class Solution: |
|  | def maxCount(self, m: int, n: int, ops: List[List[int]]) -> int: |
|  | for r,c in ops: |
|  |  |
|  | m = min(m,r) |
|  | n = min(c,n) |
|  |  |
|  | return m\*n |

Q8. Given the array nums consisting of 2n elements in the form [x1,x2,...,xn,y1,y2,...,yn].\*Return the array in the form\* [x1,y1,x2,y2,...,xn,yn].

class Solution:

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

l = 0

r = n

res = []

while l < n:

res.append(nums[l])

res.append(nums[r])

l+=1

r+=1

return res