**Find time complexity of below code blocks :**

Problem 1:

def quicksort(arr):

if len(arr) <= 1:

return arr

pivot = arr[len(arr) // 2]

left = [x for x in arr if x < pivot]

middle = [x for x in arr if x == pivot]

right = [x for x in arr if x > pivot]

return quicksort(left) + middle + quicksort(right)

Ans-> O(n logn)

Problem 2 :

def nested\_loop\_example(matrix):

rows, cols = len(matrix), len(matrix[0])

total = 0

for i in range(rows):

for j in range(cols):

total += matrix[i][j]

return total

ans-> O(n^2)

Problem 3 :

def example\_function(arr):

result = 0

for element in arr:

result += element

return result

Ans->O(n)

Problem 4 :

def longest\_increasing\_subsequence(nums):

n = len(nums)

lis = [1] \* n

for i in range(1, n):

for j in range(0, i):

if nums[i] > nums[j] and lis[i] < lis[j] + 1:

lis[i] = lis[j] + 1

return max(lis)

Ans->O(n^2)

Problem 5 :

def mysterious\_function(arr):

n = len(arr)

result = 0

for i in range(n):

for j in range(i, n):

result += arr[i] \* arr[j]

return result

Ans->O(n^2)