**Reverse of a number using recursion**

def reversenumber(number, reversed\_num=0):

if number == 0:

return reversed\_num

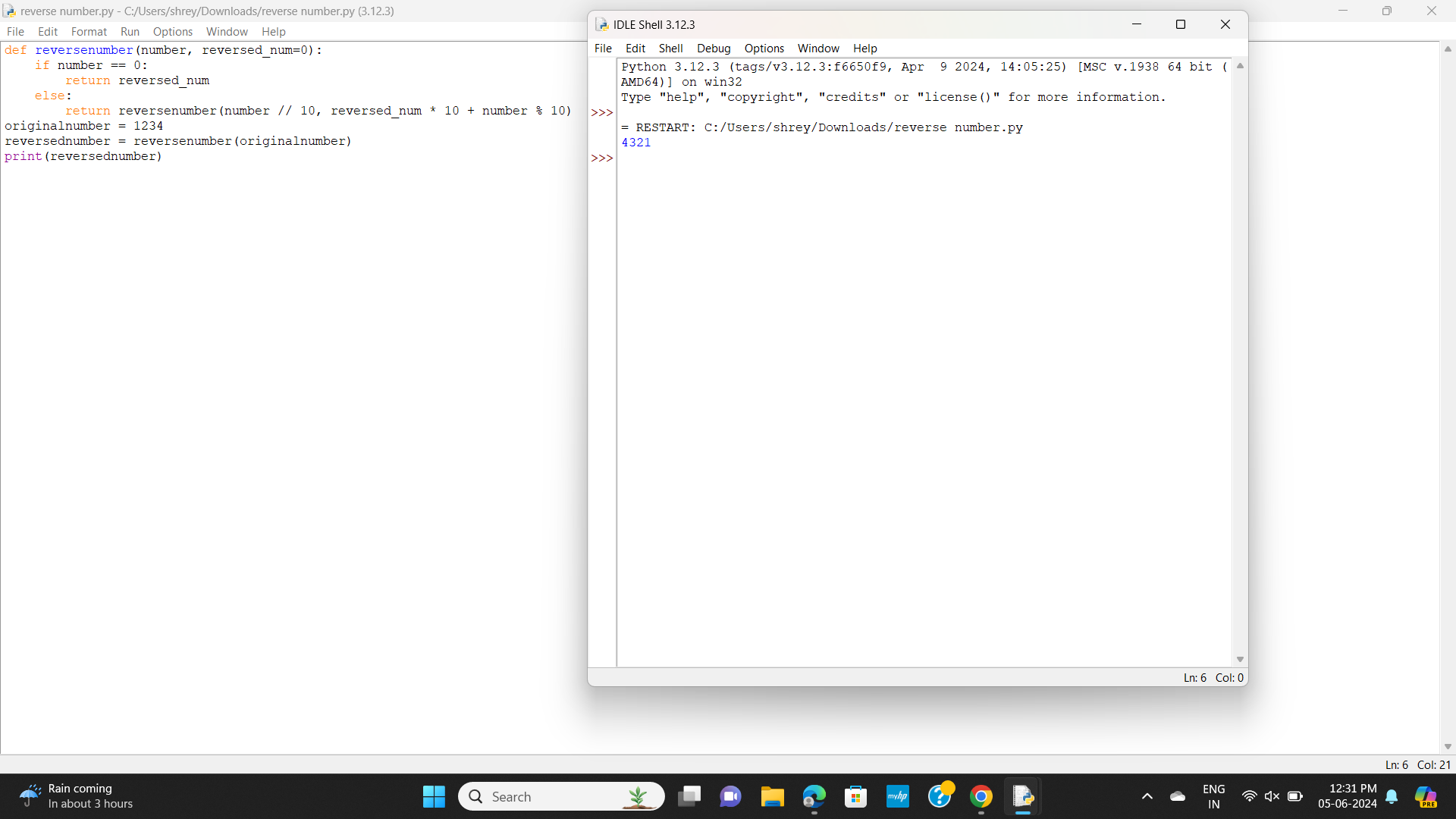
else:

return reversenumber(number // 10, reversed\_num \* 10 + number % 10)

originalnumber = 1234

reversednumber = reversenumber(originalnumber)

print(reversednumber)



**Perfect number**

num=6

Sum = 0

for i in range(1, num):

if(num % i == 0):

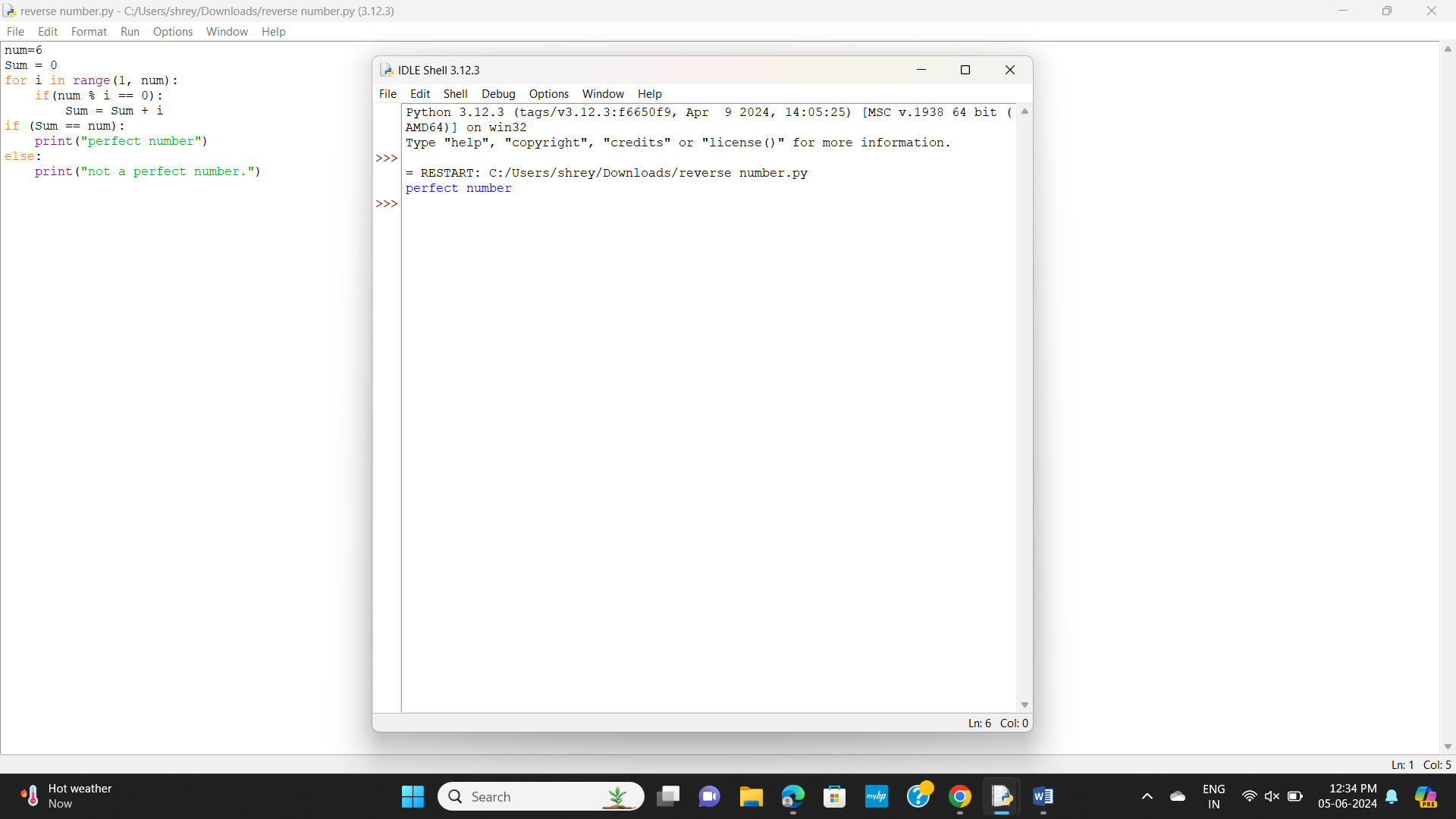
Sum = Sum + i

if (Sum == num):

print("perfect number")

else:

print("not a perfect number.")



**Time Complexity**

def constant\_time\_algorithm(arr):

return arr[0]

def linear\_time\_algorithm(arr):

total = 0

for num in arr:

total += num

return total

def quadratic\_time\_algorithm(arr):

count = 0

for i in arr:

for j in arr:

count += 1

return count

arr = list(range(10))

print("Demonstrating time complexity...")

result1 = constant\_time\_algorithm(arr)

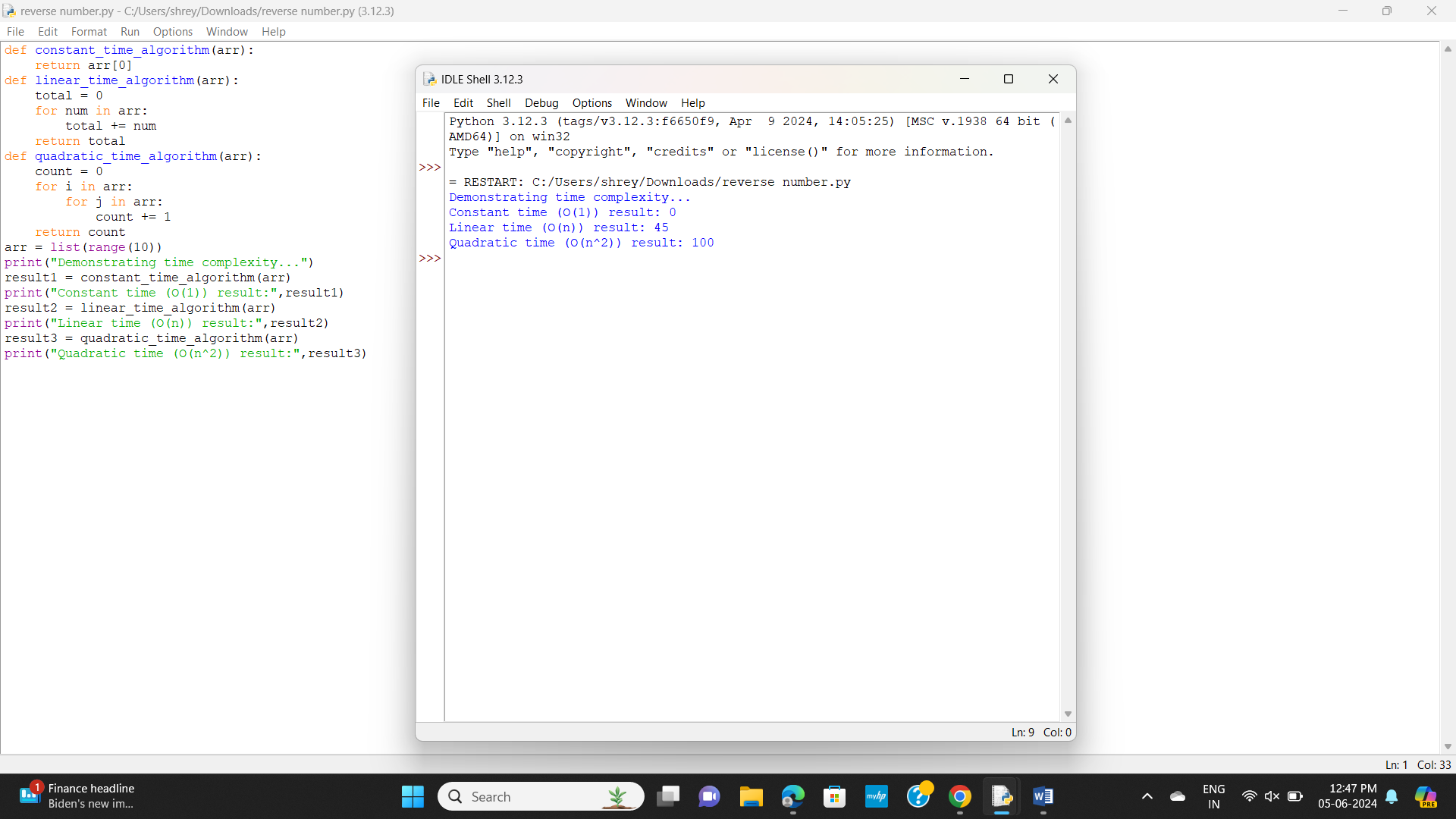
print("Constant time (O(1)) result:",result1)

result2 = linear\_time\_algorithm(arr)

print("Linear time (O(n)) result:",result2)

result3 = quadratic\_time\_algorithm(arr)

print("Quadratic time (O(n^2)) result:",result3)



**Recursive and non-recursive**

def linear(arr, target):

for i in range(len(arr)):

if arr[i] == target:

return i

return -1

def factorial(n):

if n == 0 or n == 1:

return 1

else:

return n \* factorial(n - 1)

arr = list(range(5))

target = 3

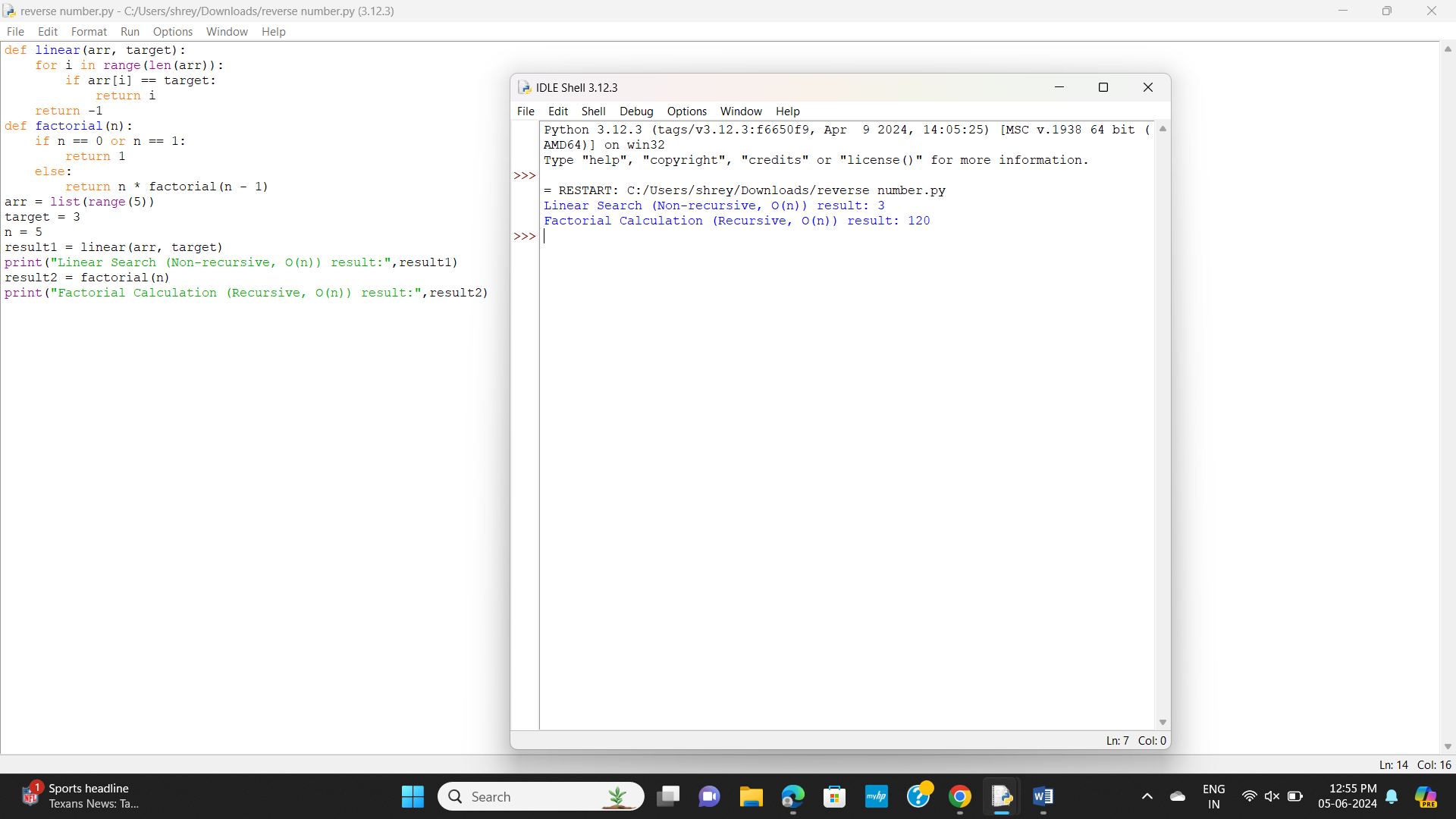
n = 5

result1 = linear(arr, target)

print("Linear Search (Non-recursive, O(n)) result:",result1)

result2 = factorial(n)

print("Factorial Calculation (Recursive, O(n)) result:",result2)



**Intersection of given 2 arrays**

def intersection(nums1, nums2):

set1 = set(nums1)

set2 = set(nums2)

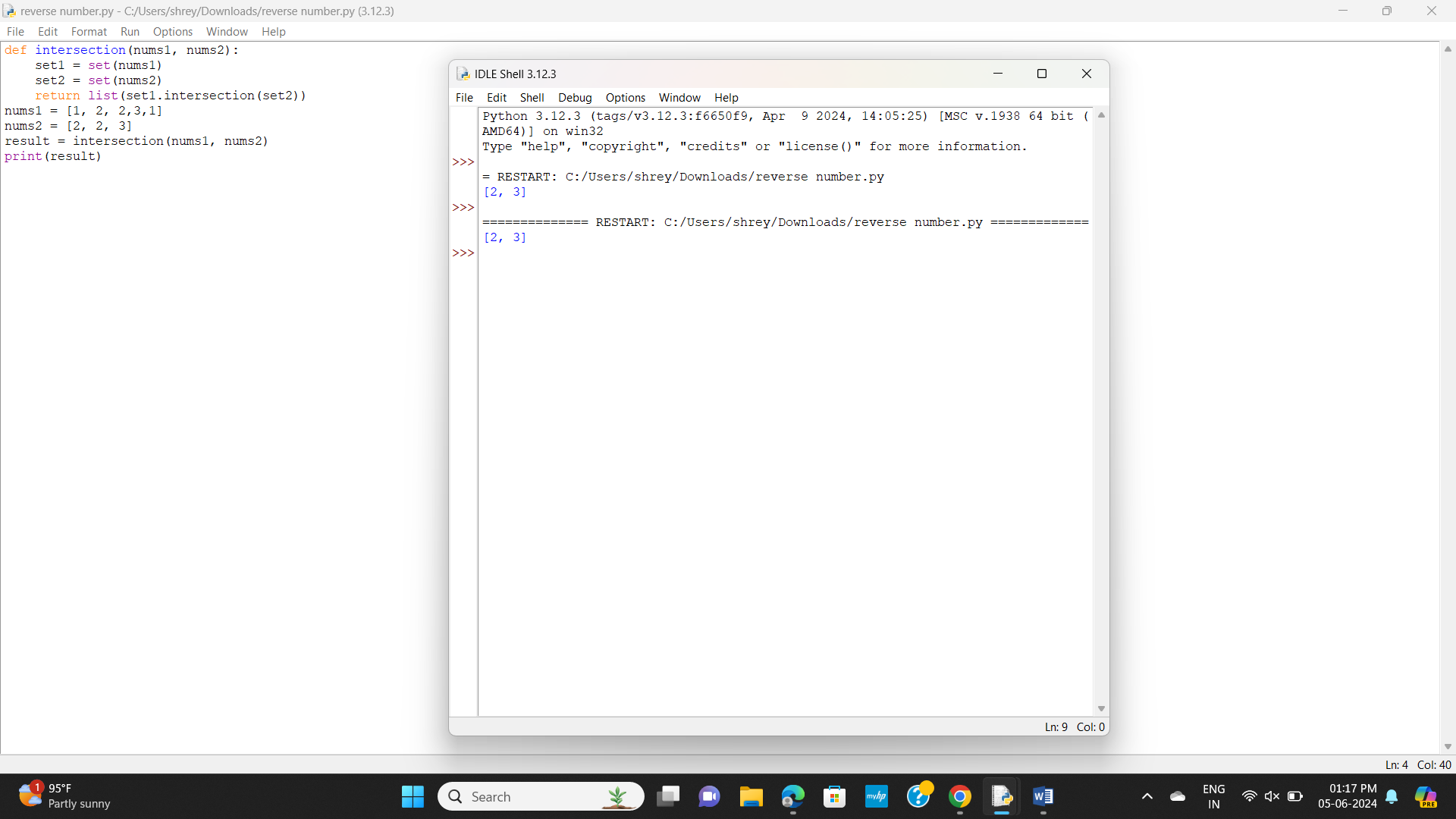
return list(set1.intersection(set2))

nums1 = [1, 2, 2,3,1]

nums2 = [2, 2, 3]

result = intersection(nums1, nums2)

print(result)

****

**Intersection of 2 arrays**

from collections import Counter

def intersection(nums1, nums2):

counts1 = Counter(nums1)

counts2 = Counter(nums2)

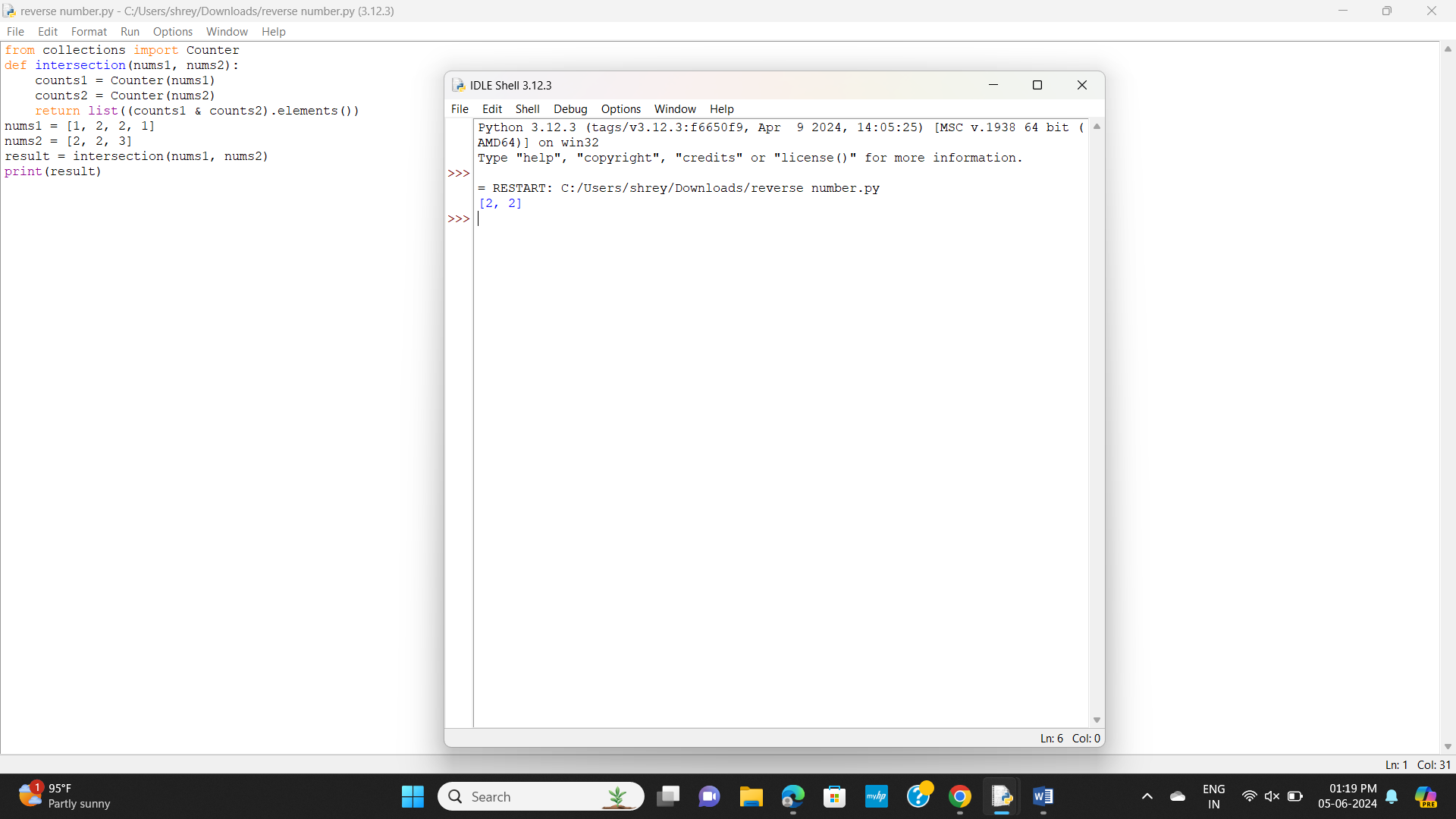
return list((counts1 & counts2).elements())

nums1 = [1, 2, 2, 1]

nums2 = [2, 2, 3]

result = intersection(nums1, nums2)

print(result)



**Sort an array**

a=[1,7,9,4,5]

n=5

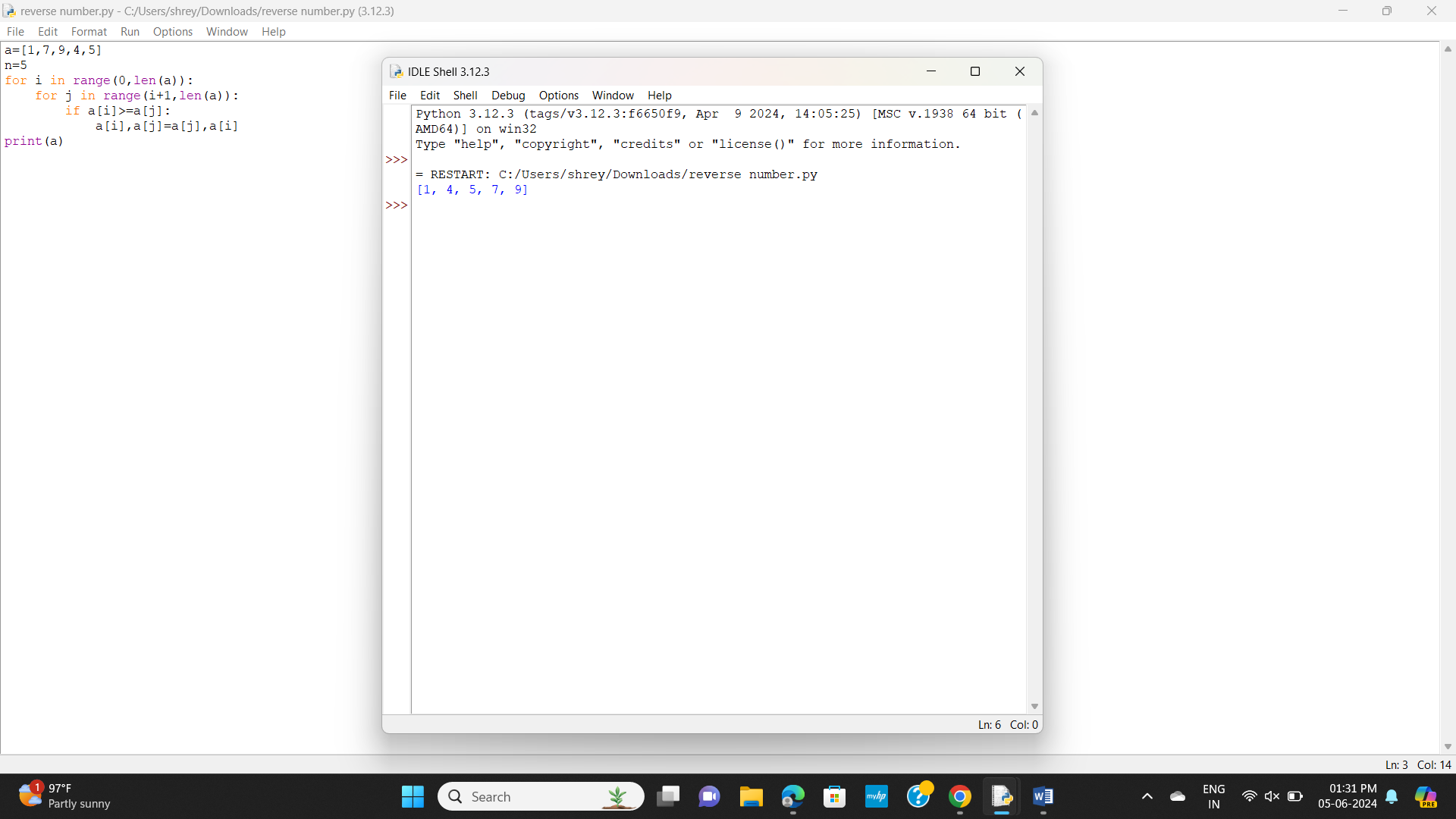
for i in range(0,len(a)):

for j in range(i+1,len(a)):

if a[i]>=a[j]:

a[i],a[j]=a[j],a[i]

print(a)



**Arrange numbers in an array in left of even numbers and right of odd numbers**

def rearrange(nums):

left, right = 0, len(nums) - 1

while left < right:

while nums[left] % 2 == 0 and left < right:

left += 1

while nums[right] % 2 == 1 and left < right:

right -= 1

if left < right:

nums[left], nums[right] = nums[right], nums[left]

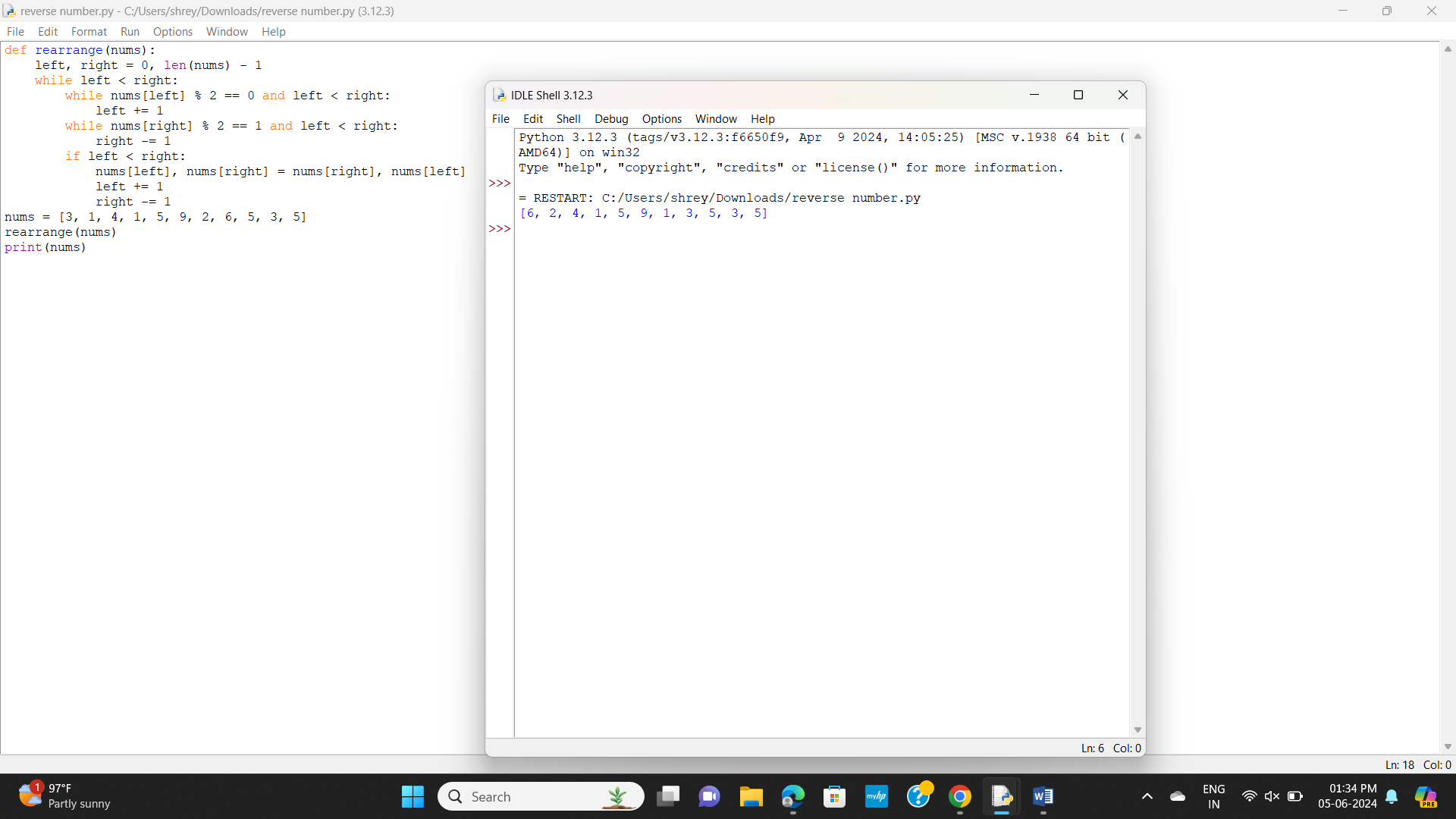
left += 1

right -= 1

nums = [3, 1, 4, 1, 5, 9, 2, 6, 5, 3, 5]

rearrange(nums)

print(nums)



**Odd and Even numbers in an array**

def sort\_array\_by\_parity(nums):

even, odd = 0, 1

while even < len(nums) and odd < len(nums):

if nums[even] % 2 != 0 and nums[odd] % 2 == 0:

nums[even], nums[odd] = nums[odd], nums[even]

if nums[even] % 2 == 0:

even += 2

if nums[odd] % 2 != 0:

odd += 2

return nums

nums = [3, 1, 4, 1, 5, 9, 2, 6, 5, 3, 5]

result = sort\_array\_by\_parity(nums)

print(result)

