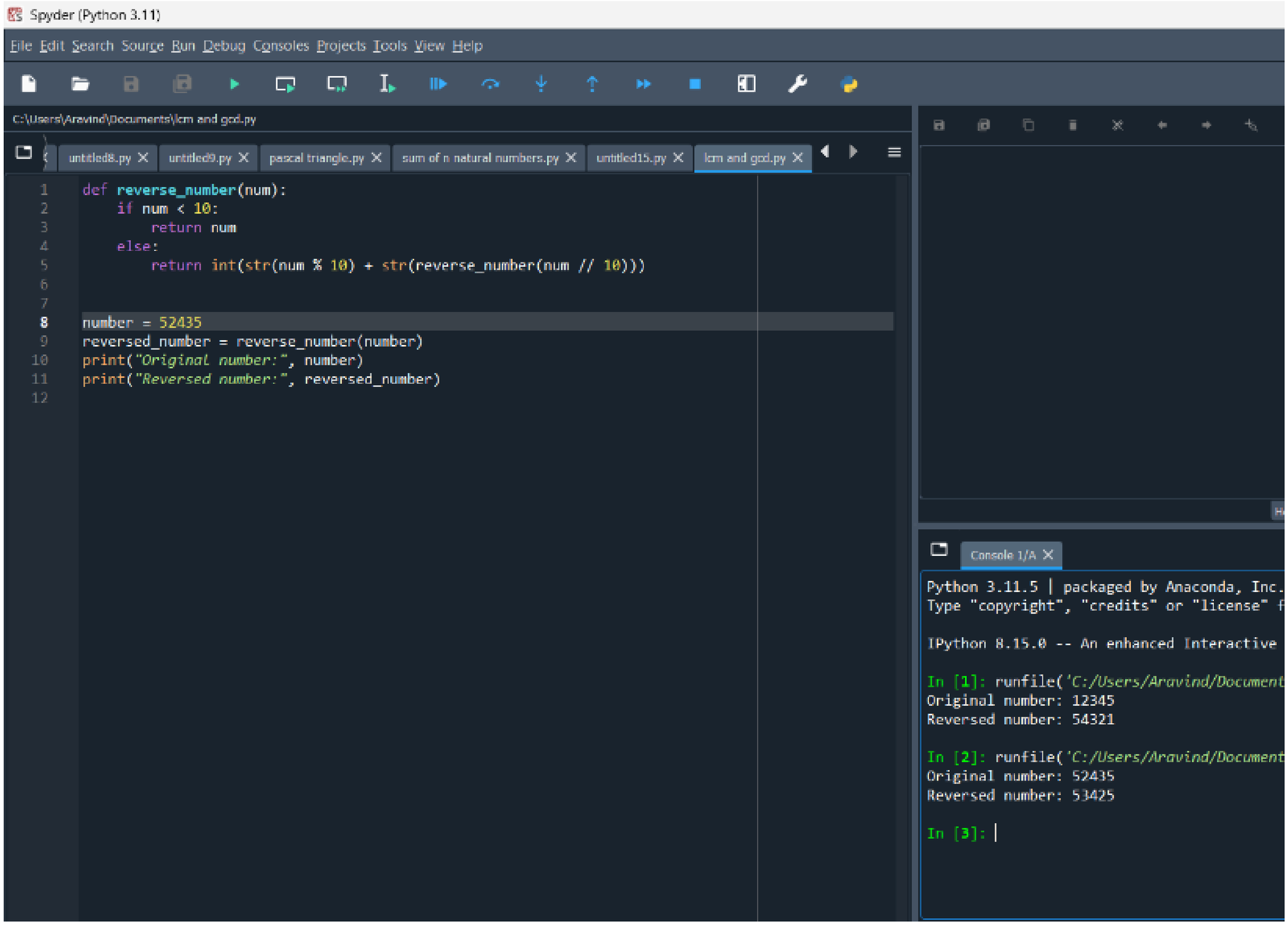
1. Write a program to find the reverse of a given number using recursive.



2. Write a program to find the perfect number

Spyder (Python 3.11)

File Edit Search Source Run Debug Consoles Projects Tools View Help

C:\Users\Aravind\Documents\lcm and gcd.py

untitled8.py | untitled9.py | pascal triangle.py | sum of n natural numbers.py | untitled15.py | lcm and gcd.py

```python
def is_perfect_number(num):
    if num <= 1:
        return False
    divisor_sum = 1
    for i in range(2, int(num**0.5) + 1):
        if num % i == 0:
            divisor_sum += i
            if i != num // i:
                divisor_sum += num // i
    return divisor_sum == num


number = 28
if is_perfect_number(number):
    print(number, "is a perfect number")
else:
    print(number, "is not a perfect number")
```

Console 2/A

```
Python 3.11.5 | packaged by Anacon
Type "copyright", "credits" or "li

IPython 8.15.0 -- An enhanced Inte

In [1]: runfile('C:/Users/Aravind/
28 is a perfect number

In [2]:
```

3. Write C program that demonstrates the usage of these notations by analyzing the time complexity of some example algorithms.

python.py - C:/Users/Aravind/AppData/Local/Programs/Python/Python312/python.py (3.12.1)

File Edit Format Run Options Window Help

```python
def constant_algo(items):
    result = items[0] * items[0]
    print(result)

def linear_algo(items):
    for item in items:
        print(item)

def quadratic_algo(items):
    for item in items:
        for i in range(len(items)):
            print(item, items[i])

def logarithmic_algo(n):
    while n > 1:
        print(n)
        n = n // 2

def exponential_algo(n):
    if n <= 1:
        return n
    else:
        return exponential_algo(n-1) + exponential_algo(n-2)

items = [1, 2, 3, 4, 5]
constant_algo(items)
linear_algo(items)
quadratic_algo(items)
logarithmic_algo(16)
exponential_algo(5)
```

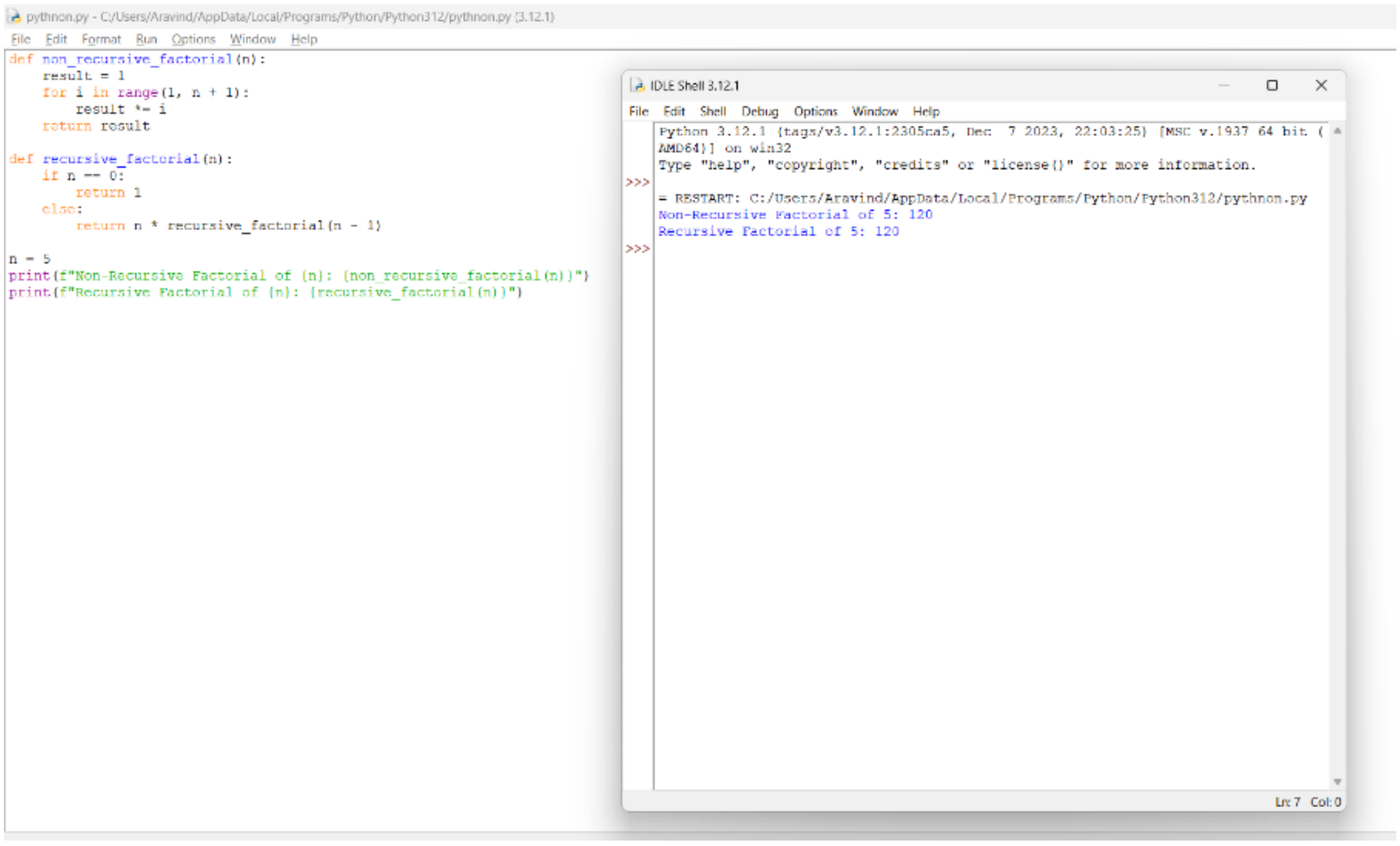
IDLE Shell 3.12.1

File Edit Shell Debug Options Window Help

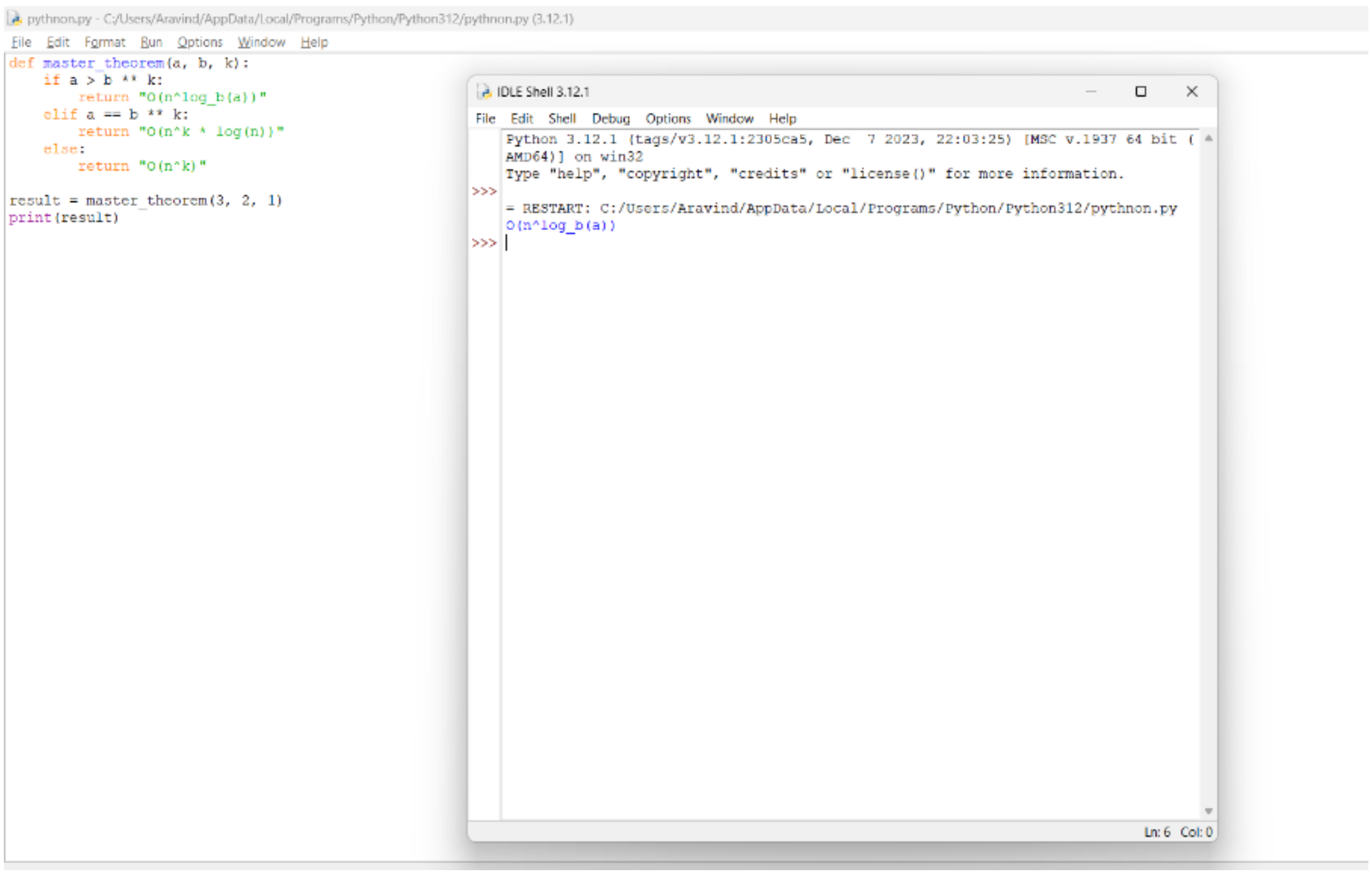
```
= RESTART: C:/Users/Aravind/AppData/Local/Programs/Python/Python312/python.py
Sorted array: [7, 56, 9, 4, 7, 2]
>>>
= RESTART: C:/Users/Aravind/AppData/Local/Programs/Python/Python312/python.py =
1
1
2
3
4
5
1 1
1 2
1 3
1 4
1 5
2 1
2 2
2 3
2 4
2 5
3 1
3 2
3 3
3 4
3 5
4 1
4 2
4 3
4 4
4 5
5 1
5 2
5 3
5 4
5 5
16
0
4
2
>>>
```

Ln: 43 Col: 0

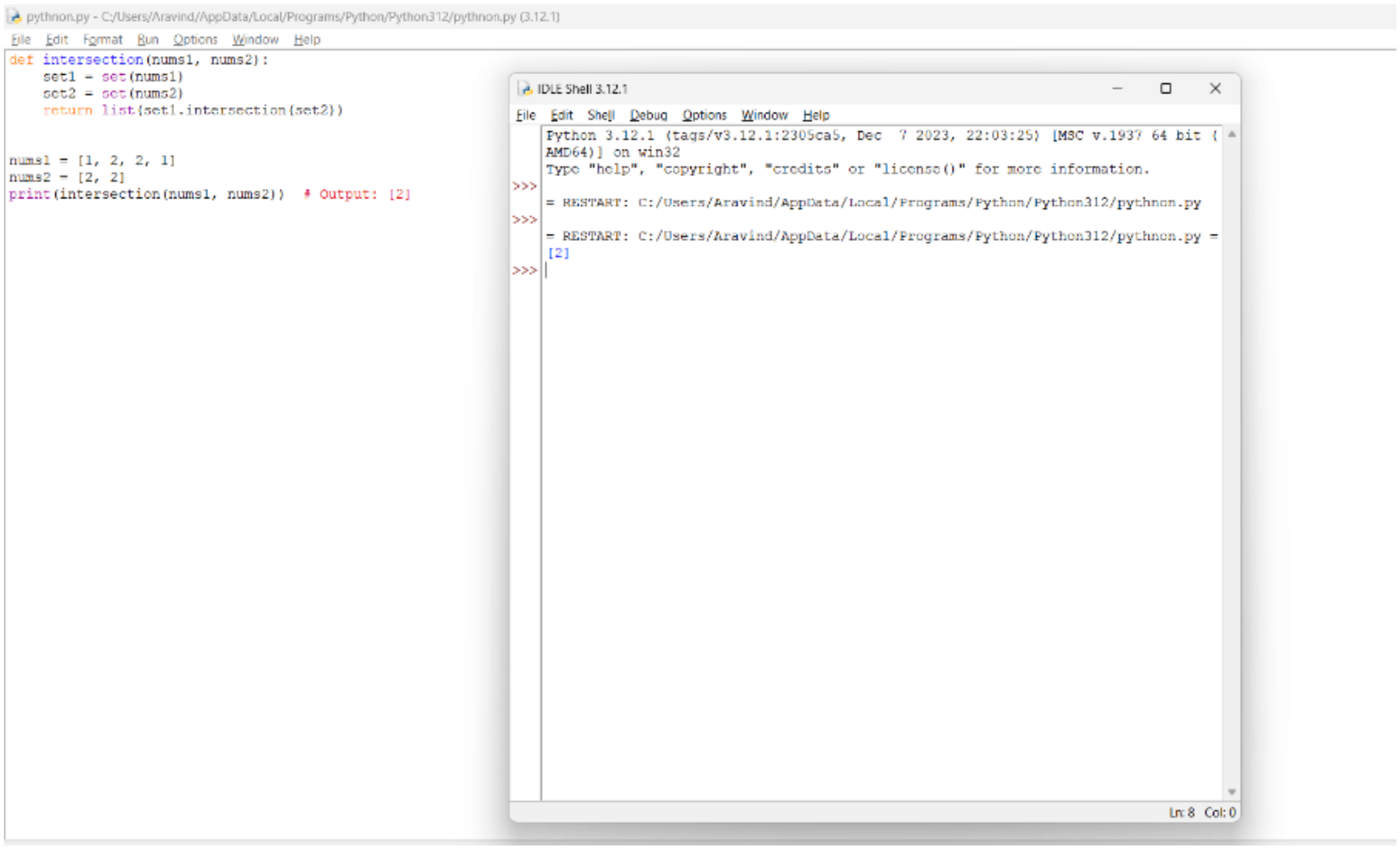
4. Write C programs that demonstrate the mathematical analysis of non-recursive and recursive algorithms



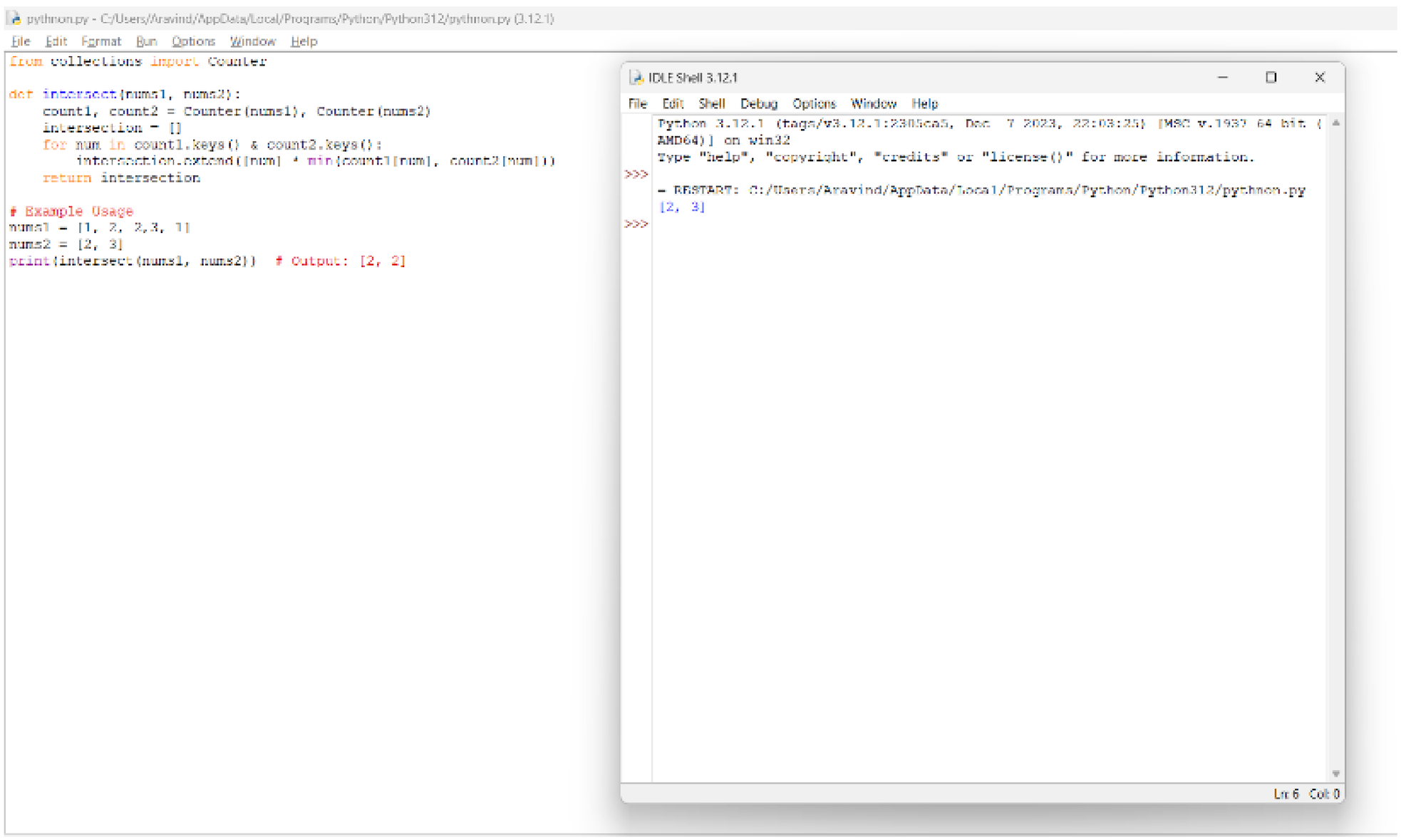
5. Write C programs for solving recurrence relations using the Master Theorem, Substitution Method, and Iteration Method will demonstrate how to calculate the time complexity of an example recurrence relation using the specified technique.



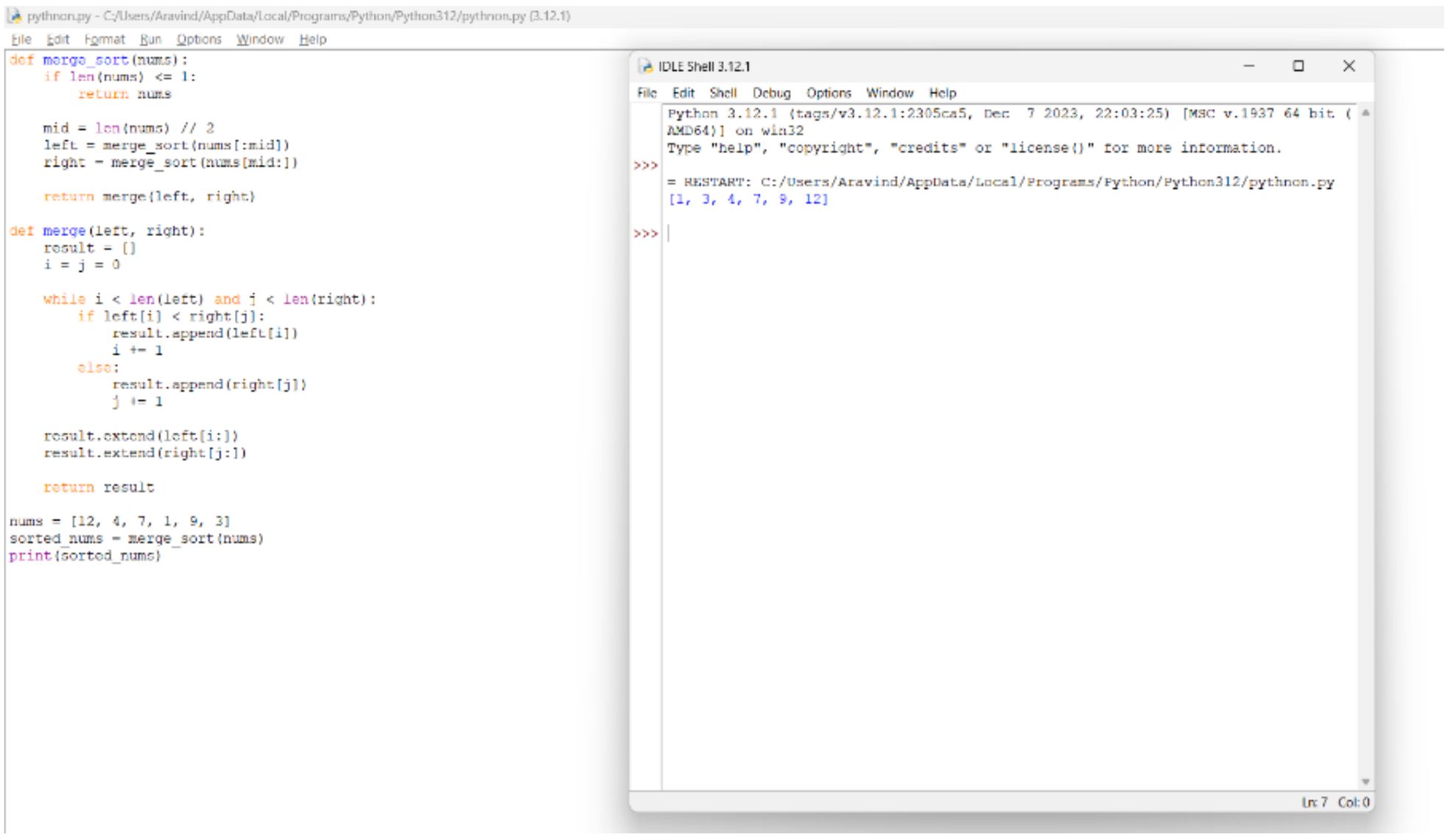
6. Given two integer arrays nums1 and nums2, return an array of their Intersection. Each element in the result must be unique and you may return the result in any order.



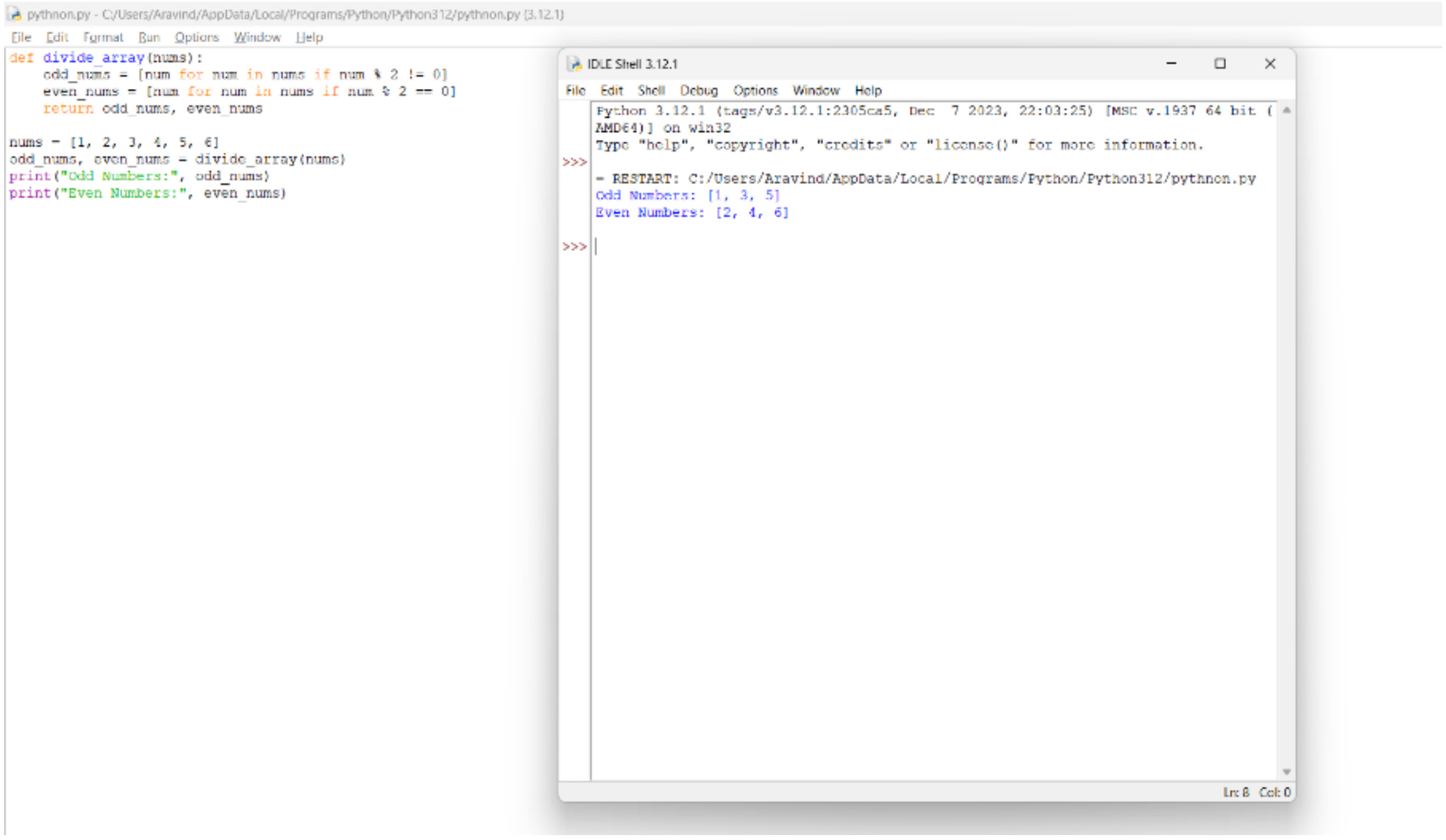
7. Given two integer arrays nums1 and nums2, return an array of their intersection. Each element in the result must appear as many times as it shows in both arrays and you may return the result in any order.



8. Given an array of integers nums, sort the array in ascending order and return it. You must solve the problem without using any built-in functions in O(nlog(n)) time complexity and with the smallest space complexity possible.



9. Given an array of integers nums, half of the integers in nums are odd, and the other half are even.



10. Sort the array so that whenever nums[i] is odd, i is odd, and whenever nums[i] is even, i is even. Return any answer array that satisfies this condition.

pythnon.py - C:/Users/Aravind/AppData/Local/Programs/Python/Python312/pythnon.py (3.12.1)

File Edit Format Run Options Window Help

```python
def odd_even_sort(nums):
    nums.sort(key=lambda x: (x % 2, x % 4))
    return nums

arr = [4, 2, 5, 7, 6, 1, 3]
sorted_arr = odd_even_sort(arr)
print(sorted_arr)
```

IDLE Shell 3.12.1

File Edit Shell Debug Options Window Help

```
Python 3.12.1 (tags/v3.12.1:2305ca5, Dec  7 2023, 22:03:25) [MSC v.1937 64 bit (
AMD64)] on win32
Type "help", "copyright", "credits" or "license()" for more information.
>>>
= RESTART: C:/Users/Aravind/AppData/Local/Programs/Python/Python312/pythnon.py
[4, 2, 6, 5, 1, 7, 3]
>>>
```

Ln: 6 Col: 0