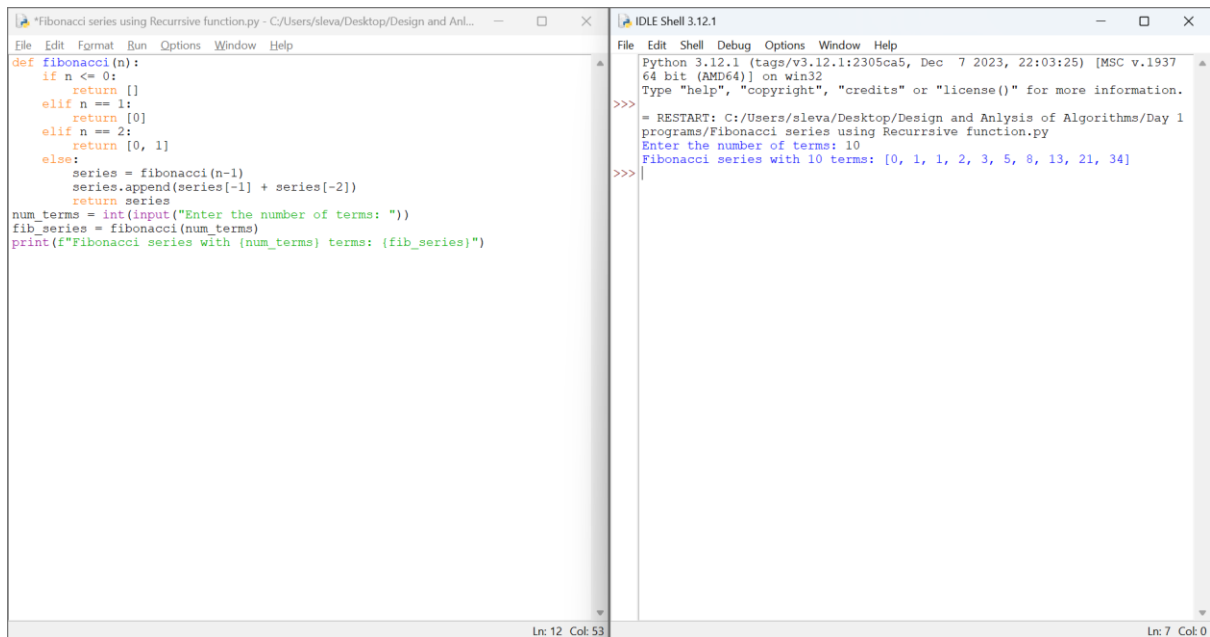


## 1. Fibonacci Series using recursion.



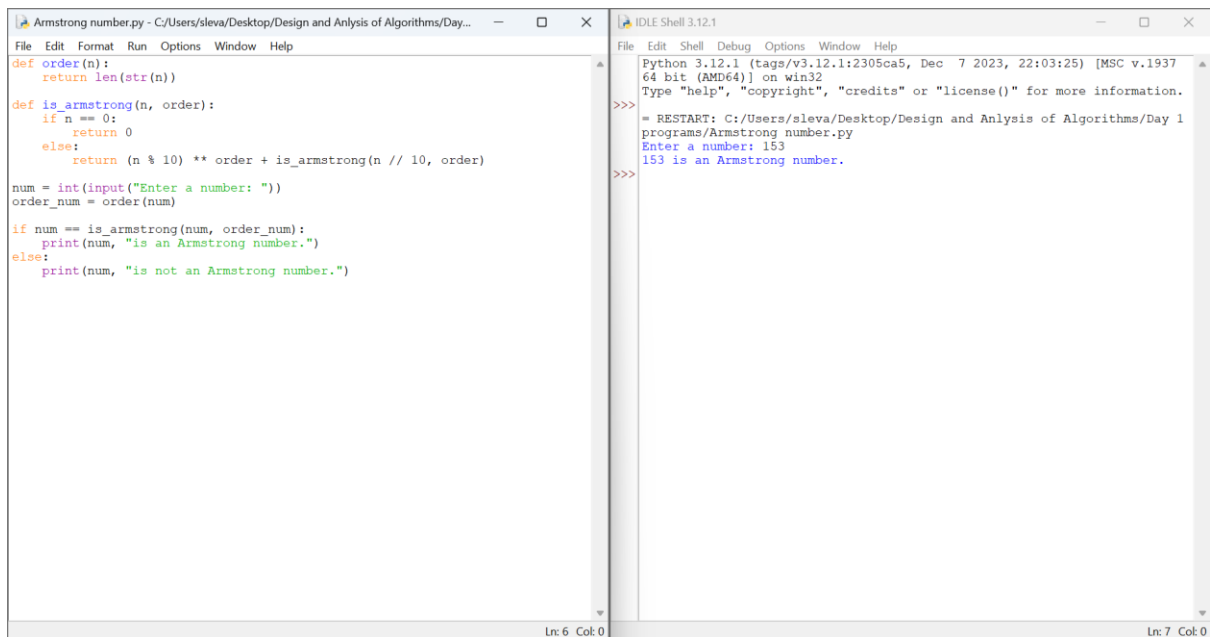
The screenshot shows an IDE with two windows. The left window, titled "Fibonacci series using Recursive function.py", contains the following Python code:

```
def fibonacci(n):
    if n <= 0:
        return []
    elif n == 1:
        return [0]
    elif n == 2:
        return [0, 1]
    else:
        series = fibonacci(n-1)
        series.append(series[-1] + series[-2])
        return series
num_terms = int(input("Enter the number of terms: "))
fib_series = fibonacci(num_terms)
print(f"Fibonacci series with {num_terms} terms: {fib_series}")
```

The right window, titled "IDLE Shell 3.12.1", shows the execution output:

```
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/sleval/Desktop/Design and Anlysis of Algorithms/Day 1
programs/Fibonacci series using Recursive function.py
Enter the number of terms: 10
Fibonacci series with 10 terms: [0, 1, 1, 2, 3, 5, 8, 13, 21, 34]
>>>
```

## 2. Armstrong or not using recursive function.



The screenshot shows an IDE with two windows. The left window, titled "Armstrong number.py", contains the following Python code:

```
def order(n):
    return len(str(n))

def is_armstrong(n, order):
    if n == 0:
        return 0
    else:
        return (n % 10) ** order + is_armstrong(n // 10, order)

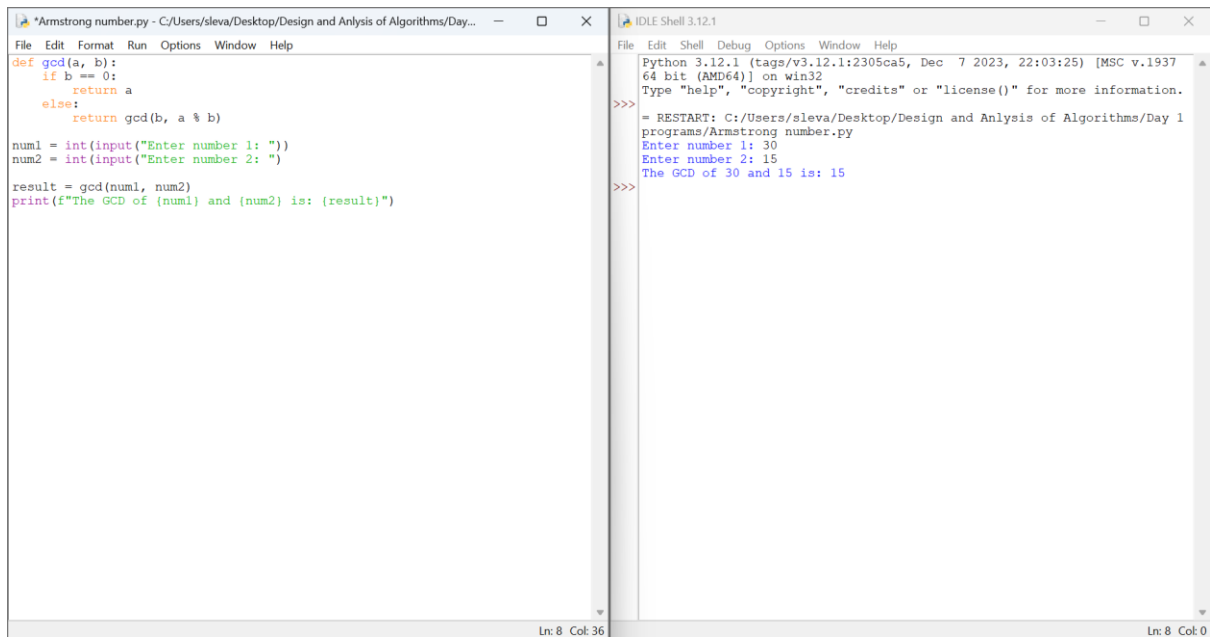
num = int(input("Enter a number: "))
order_num = order(num)

if num == is_armstrong(num, order_num):
    print(num, "is an Armstrong number.")
else:
    print(num, "is not an Armstrong number.")
```

The right window, titled "IDLE Shell 3.12.1", shows the execution output:

```
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/sleval/Desktop/Design and Anlysis of Algorithms/Day 1
programs/Armstrong number.py
Enter a number: 153
153 is an Armstrong number.
>>>
```

### 3. GCD of Two numbers using Recursion



The screenshot shows an IDE with two windows. The left window, titled "Armstrong number.py", contains the following Python code:

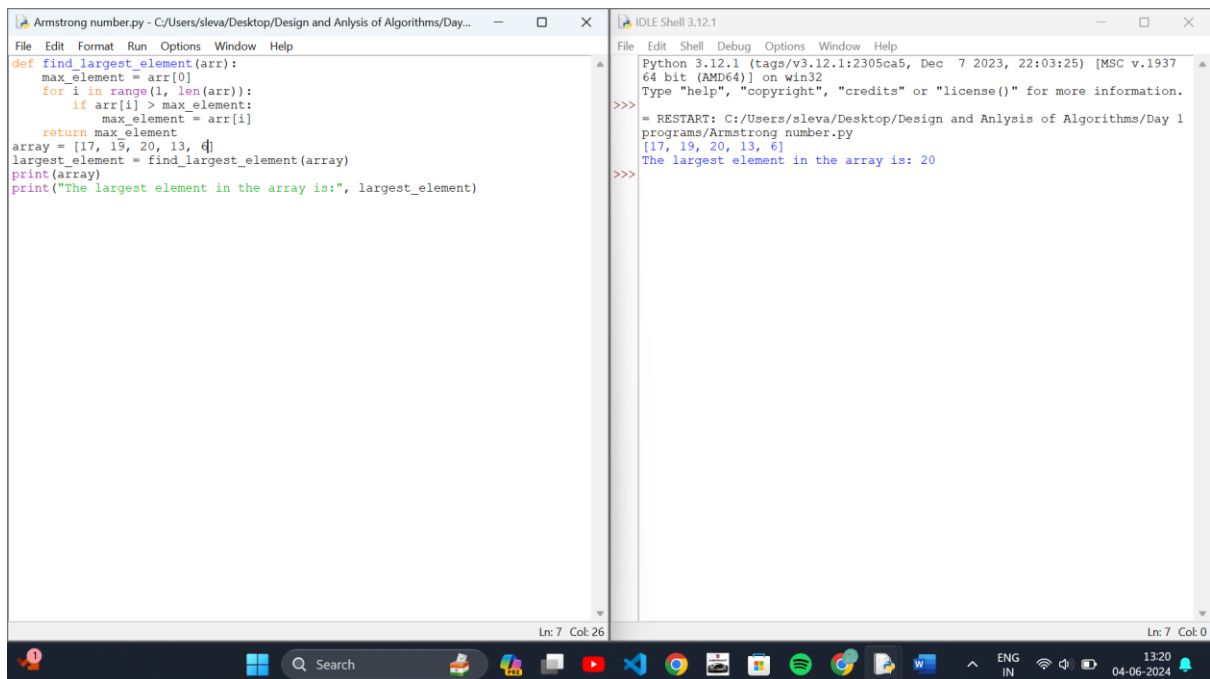
```
def gcd(a, b):  
    if b == 0:  
        return a  
    else:  
        return gcd(b, a % b)  
  
num1 = int(input("Enter number 1: "))  
num2 = int(input("Enter number 2: "))  
  
result = gcd(num1, num2)  
print(f"The GCD of {num1} and {num2} is: {result}")
```

The right window, titled "IDLE Shell 3.12.1", shows the execution output:

```
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/slewa/Desktop/Design and Anlysis of Algorithms/Day 1  
programs/Armstrong number.py  
Enter number 1: 30  
Enter number 2: 15  
The GCD of 30 and 15 is: 15  
>>>
```

The status bar at the bottom indicates "Ln: 8 Col: 36" for the left window and "Ln: 8 Col: 0" for the right window.

### 4. Largest Element in an Array



The screenshot shows an IDE with two windows. The left window, titled "Armstrong number.py", contains the following Python code:

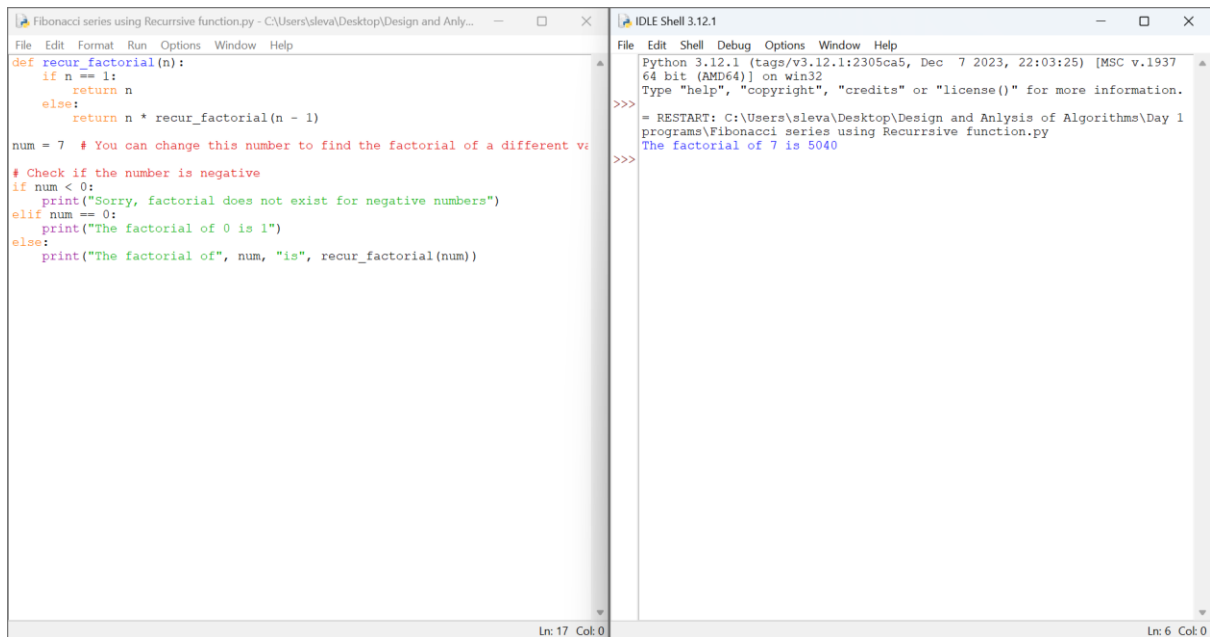
```
def find_largest_element(arr):  
    max_element = arr[0]  
    for i in range(1, len(arr)):  
        if arr[i] > max_element:  
            max_element = arr[i]  
    return max_element  
  
array = [17, 19, 20, 13, 6]  
largest_element = find_largest_element(array)  
print(array)  
print("The largest element in the array is:", largest_element)
```

The right window, titled "IDLE Shell 3.12.1", shows the execution output:

```
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/slewa/Desktop/Design and Anlysis of Algorithms/Day 1  
programs/Armstrong number.py  
[17, 19, 20, 13, 6]  
The largest element in the array is: 20  
>>>
```

The status bar at the bottom indicates "Ln: 7 Col: 26" for the left window and "Ln: 7 Col: 0" for the right window. The Windows taskbar is visible at the bottom of the screen.

## 5. Factorial of a number using recursion

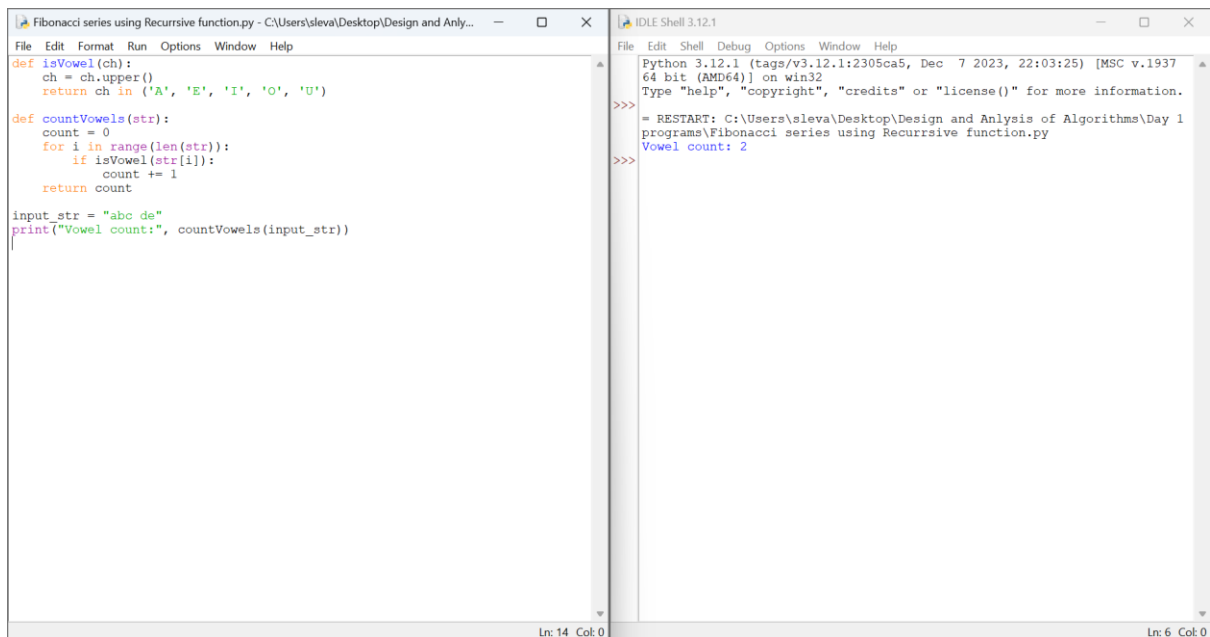


```
Fibonacci series using Recursive function.py - C:\Users\sleval\Desktop\Design and Anly...
File Edit Format Run Options Window Help
def recur_factorial(n):
    if n == 1:
        return n
    else:
        return n * recur_factorial(n - 1)

num = 7 # You can change this number to find the factorial of a different v:
# Check if the number is negative
if num < 0:
    print("Sorry, factorial does not exist for negative numbers")
elif num == 0:
    print("The factorial of 0 is 1")
else:
    print("The factorial of", num, "is", recur_factorial(num))

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\sleval\Desktop\Design and Anlysis of Algorithms\Day 1
programs\Fibonacci series using Recursive function.py
The factorial of 7 is 5040
>>>
```

## 6. copy one string to another using recursion



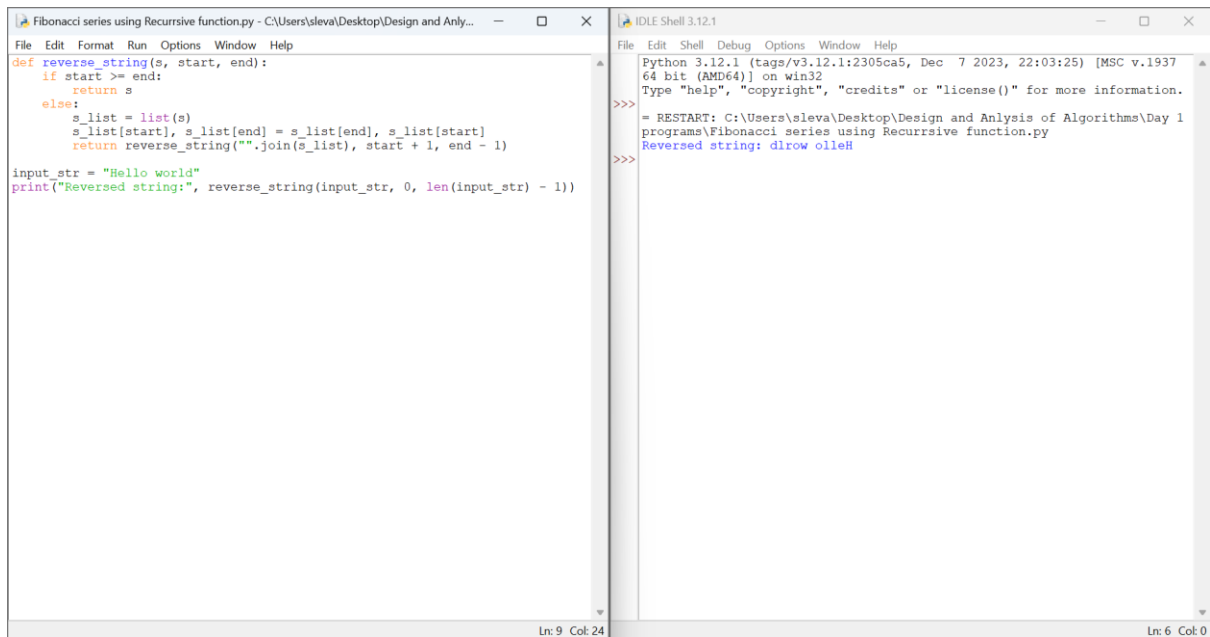
```
Fibonacci series using Recursive function.py - C:\Users\sleval\Desktop\Design and Anly...
File Edit Format Run Options Window Help
def isVowel(ch):
    ch = ch.upper()
    return ch in ('A', 'E', 'I', 'O', 'U')

def countVowels(str):
    count = 0
    for i in range(len(str)):
        if isVowel(str[i]):
            count += 1
    return count

input_str = "abc de"
print("Vowel count:", countVowels(input_str))

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\sleval\Desktop\Design and Anlysis of Algorithms\Day 1
programs\Fibonacci series using Recursive function.py
Vowel count: 2
>>>
```

## 7. Reverse a string using Recursion



```
Fibonacci series using Recursive function.py - C:\Users\sleval\Desktop\Design and Anyl...
File Edit Format Run Options Window Help
def reverse_string(s, start, end):
    if start >= end:
        return s
    else:
        s_list = list(s)
        s_list[start], s_list[end] = s_list[end], s_list[start]
        return reverse_string("".join(s_list), start + 1, end - 1)

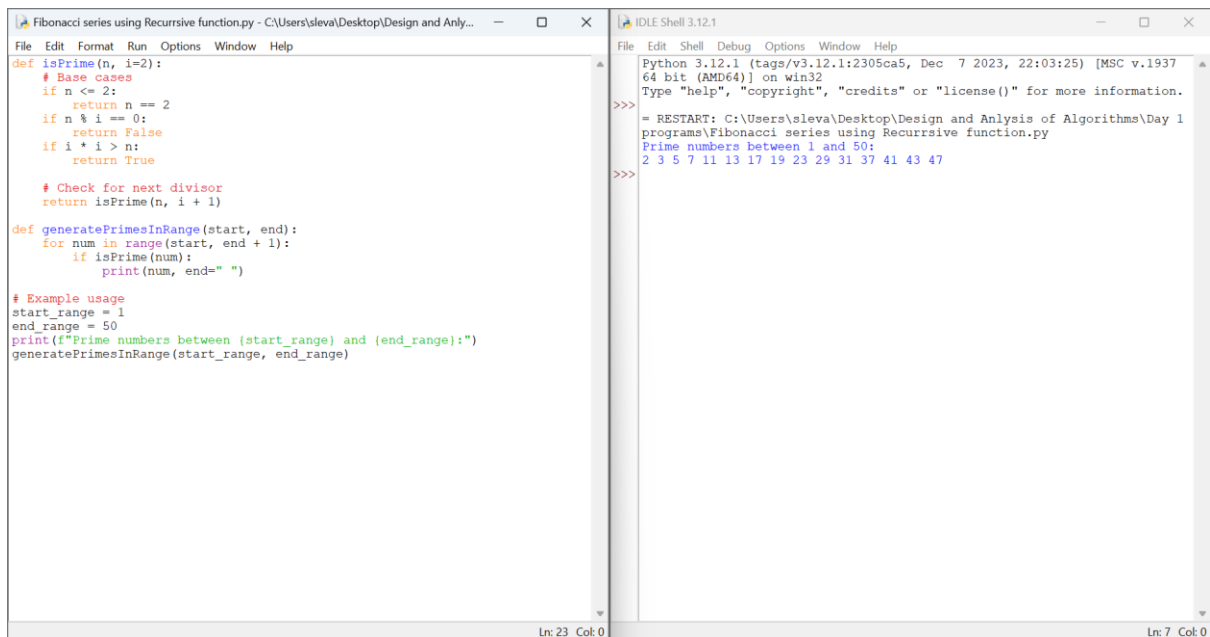
input_str = "Hello world"
print("Reversed string:", reverse_string(input_str, 0, len(input_str) - 1))

Ln: 9 Col: 24

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\sleval\Desktop\Design and Anyl... of Algorithms\Day 1
programs\Fibonacci series using Recursive function.py
Reversed string: dlrow olleH
>>>

Ln: 6 Col: 0
```

## 8. Generate all the prime numbers using recursion



```
Fibonacci series using Recursive function.py - C:\Users\sleval\Desktop\Design and Anyl...
File Edit Format Run Options Window Help
def isPrime(n, i=2):
    # Base cases
    if n <= 2:
        return n == 2
    if n % i == 0:
        return False
    if i * i > n:
        return True

    # Check for next divisor
    return isPrime(n, i + 1)

def generatePrimesInRange(start, end):
    for num in range(start, end + 1):
        if isPrime(num):
            print(num, end=" ")

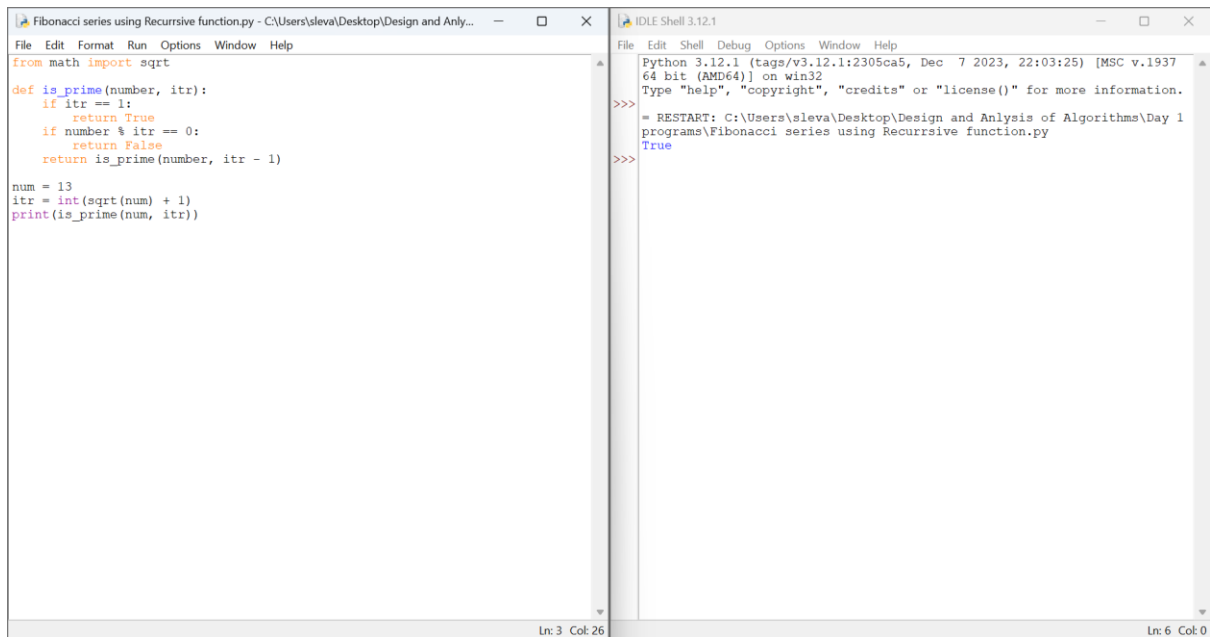
# Example usage
start_range = 1
end_range = 50
print(f"Prime numbers between {start_range} and {end_range}:")
generatePrimesInRange(start_range, end_range)

Ln: 23 Col: 0

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\sleval\Desktop\Design and Anyl... of Algorithms\Day 1
programs\Fibonacci series using Recursive function.py
Prime numbers between 1 and 50:
2 3 5 7 11 13 17 19 23 29 31 37 41 43 47
>>>

Ln: 7 Col: 0
```

## 9. Check a number is a prime number or not using recursion.



The screenshot shows a Python IDE with two windows. The left window, titled 'Fibonacci series using Recursive function.py', contains the following code:

```
from math import sqrt

def is_prime(number, itr):
    if itr == 1:
        return True
    if number % itr == 0:
        return False
    return is_prime(number, itr - 1)

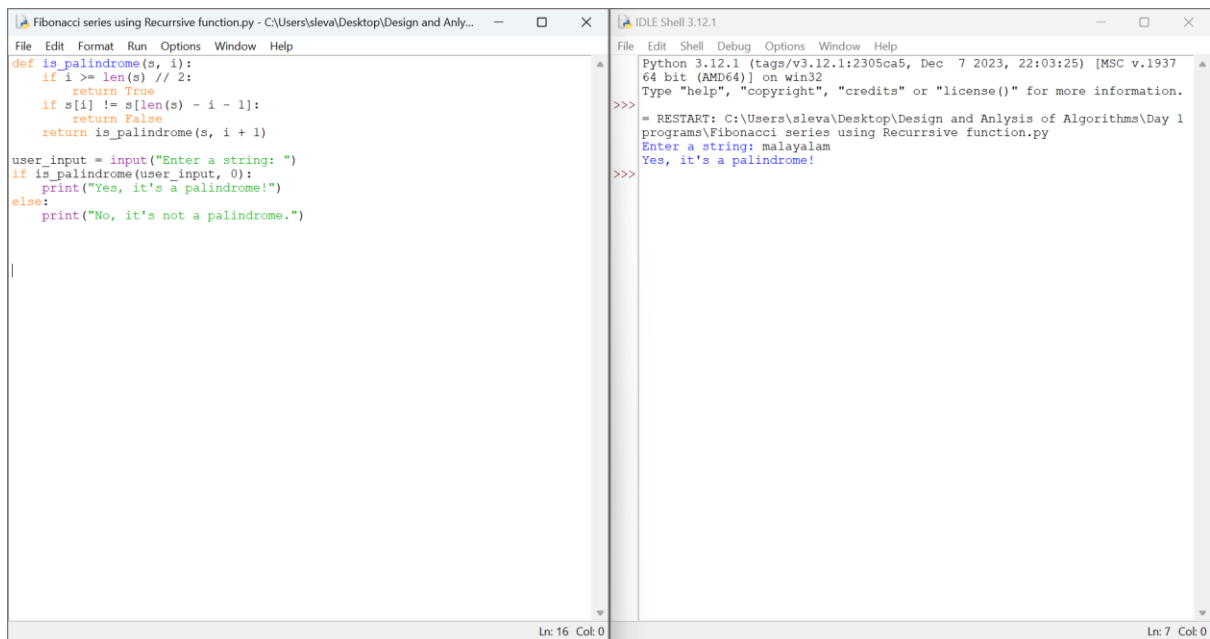
num = 13
itr = int(sqrt(num) + 1)
print(is_prime(num, itr))
```

The right window, titled 'IDLE Shell 3.12.1', shows the output of the program:

```
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\sleval\Desktop\Design and Anlysis of Algorithms\Day 1
programs\Fibonacci series using Recursive function.py
True
>>>
```

The status bar at the bottom indicates 'Ln: 3 Col: 26' for the left window and 'Ln: 6 Col: 0' for the right window.

## 10. Given String is Palindrome or not using recursion



The screenshot shows a Python IDE with two windows. The left window, titled 'Fibonacci series using Recursive function.py', contains the following code:

```
def is_palindrome(s, i):
    if i >= len(s) // 2:
        return True
    if s[i] != s[len(s) - i - 1]:
        return False
    return is_palindrome(s, i + 1)

user_input = input("Enter a string: ")
if is_palindrome(user_input, 0):
    print("Yes, it's a palindrome!")
else:
    print("No, it's not a palindrome.")
```

The right window, titled 'IDLE Shell 3.12.1', shows the output of the program:

```
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\sleval\Desktop\Design and Anlysis of Algorithms\Day 1
programs\Fibonacci series using Recursive function.py
Enter a string: malayalam
Yes, it's a palindrome!
>>>
```

The status bar at the bottom indicates 'Ln: 16 Col: 0' for the left window and 'Ln: 7 Col: 0' for the right window.