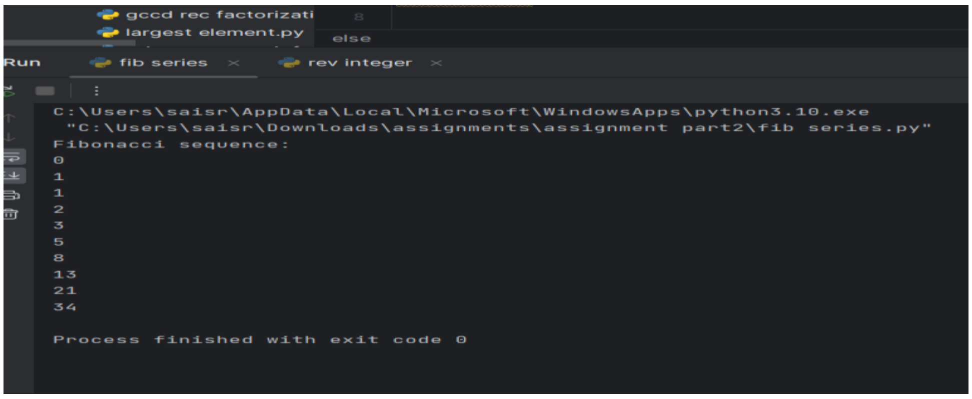
ASSIGNMENT-1

1. Write a program to Print Fibonacci Series using recursion.

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

nterms = 10

if nterms <= 0:
   print("Plese enter a positive integer")
else:
   print("Fibonacci sequence:")
   for i in range(nterms):
       print(recur_fibo(i))
```

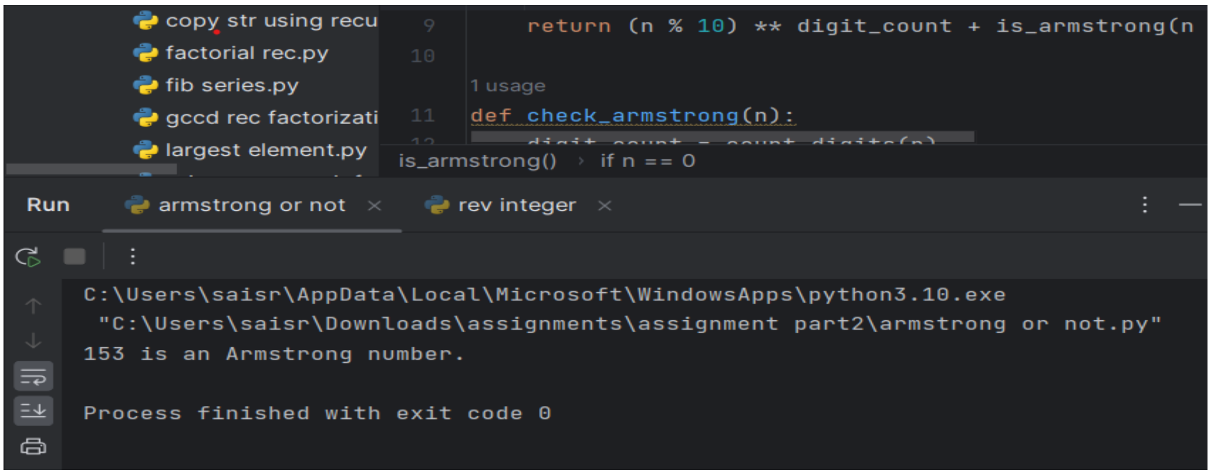


```
C:\Users\saisr\AppData\Local\Microsoft\WindowsApps\python3.10.exe
 "C:\Users\saisr\Downloads\assignments\assignment part2\fib series.py"
Fibonacci sequence:
0
1
1
2
3
5
8
13
21
34

Process finished with exit code 0
```

2. Write a program to check the given no is Armstrong or not using recursive function.

```
def count_digits(n):
    if n == 0:
        return 0
    return 1 + count_digits(n // 10)

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

def check_armstrong(n):
    digit_count = count_digits(n)
    sum_of_powers = is_armstrong(n, digit_count)
    return sum_of_powers == n

# Example usage:
num = 153
if check_armstrong(num):
    print(num, "is an Armstrong number.")
```

```python
else:
    print(num, "is not an Armstrong number.")
```



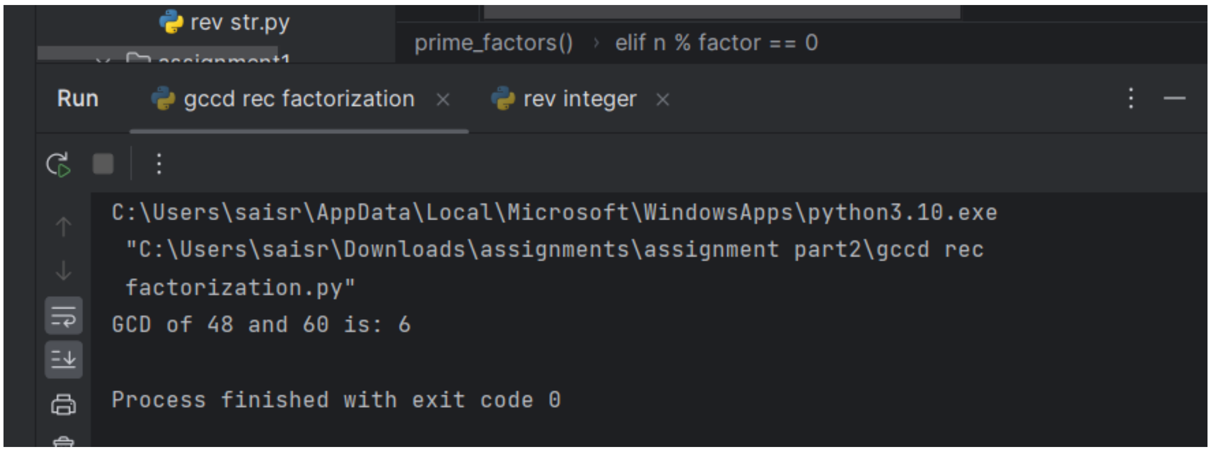
```python
    return (n % 10) ** digit_count + is_armstrong(n

def check_armstrong(n):
```

```
C:\Users\saisr\AppData\Local\Microsoft\WindowsApps\python3.10.exe
 "C:\Users\saisr\Downloads\assignments\assignment part2\armstrong or not.py"
153 is an Armstrong number.

Process finished with exit code 0
```

3.   Write a program to find the GCD of two numbers using recursive factorization

```python
def prime_factors(n, factor=2):
    if n <= 1:
        return []
    elif n % factor == 0:
        return [factor] + prime_factors(n // factor, factor)
    else:
        return prime_factors(n, factor + 1)

def gcd_recursive(a, b):
    if b == 0:
        return a
    return gcd_recursive(b, a % b)

def gcd(a, b):
    # Get prime factors of both numbers
    factors_a = prime_factors(a)
    factors_b = prime_factors(b)

    # Find common factors
    common_factors = set(factors_a) & set(factors_b)

    # Calculate GCD by multiplying common factors
    result = 1
    for factor in common_factors:
        result *= factor

    return result

# Example usage:
num1 = 48
```

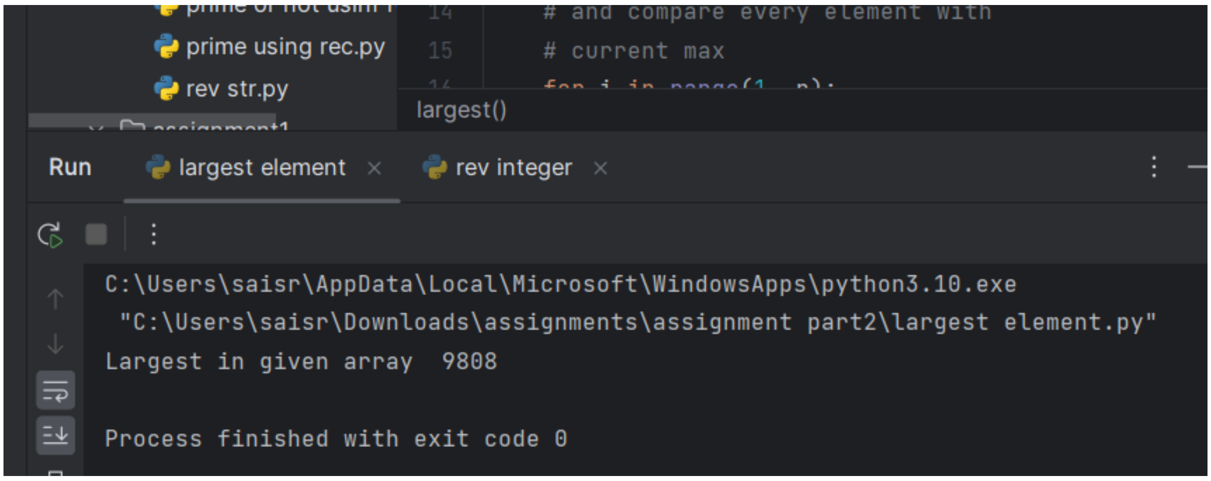
```
num2 = 60
print("GCD of", num1, "and", num2, "is:", gcd(num1, num2))   # Output: 12
```



```
C:\Users\saisr\AppData\Local\Microsoft\WindowsApps\python3.10.exe
 "C:\Users\saisr\Downloads\assignments\assignment part2\gccd rec
 factorization.py"
GCD of 48 and 60 is: 6

Process finished with exit code 0
```

4. Write a program to get the largest element of an array

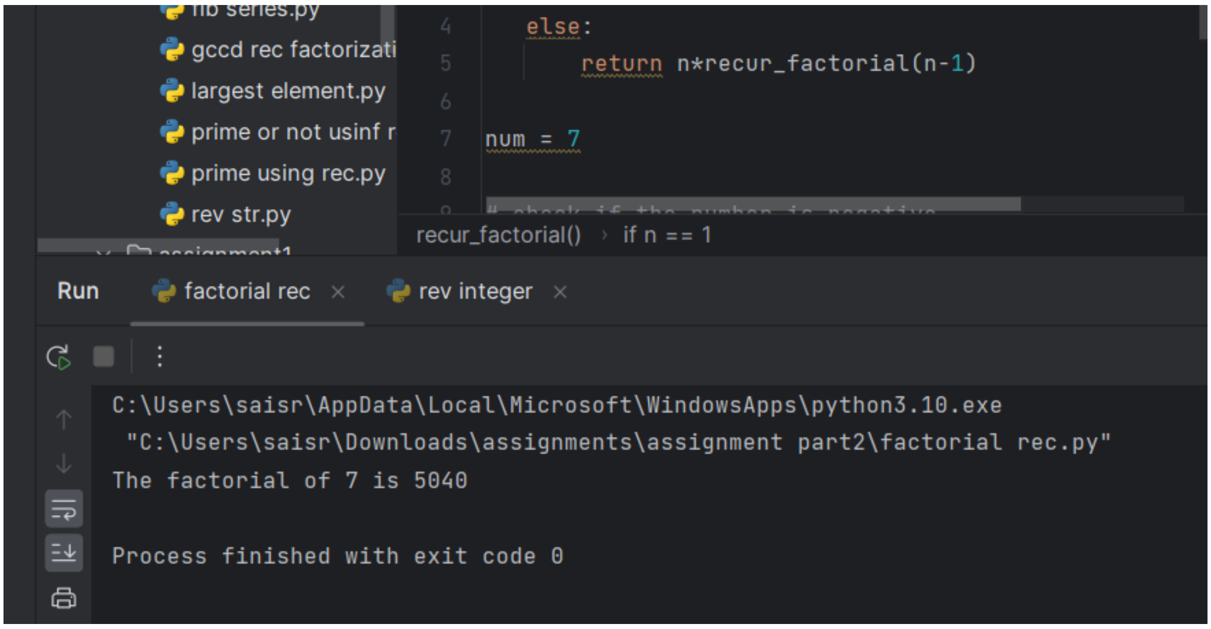
```
# Python3 program to find maximum
# in arr[] of size n

# python function to find maximum
# in arr[] of size n


def largest(arr, n):

    # Initialize maximum element
    max = arr[0]

    # Traverse array elements from second
    # and compare every element with
    # current max
    for i in range(1, n):
        if arr[i] > max:
            max = arr[i]
    return max


# Driver Code
arr = [10, 324, 45, 90, 9808]
n = len(arr)
Ans = largest(arr, n)
print("Largest in given array ", Ans)
```



```
C:\Users\saisr\AppData\Local\Microsoft\WindowsApps\python3.10.exe
 "C:\Users\saisr\Downloads\assignments\assignment part2\largest element.py"
Largest in given array  9808

Process finished with exit code 0
```

5.Write a program to find the Factorial of a number using recursion.

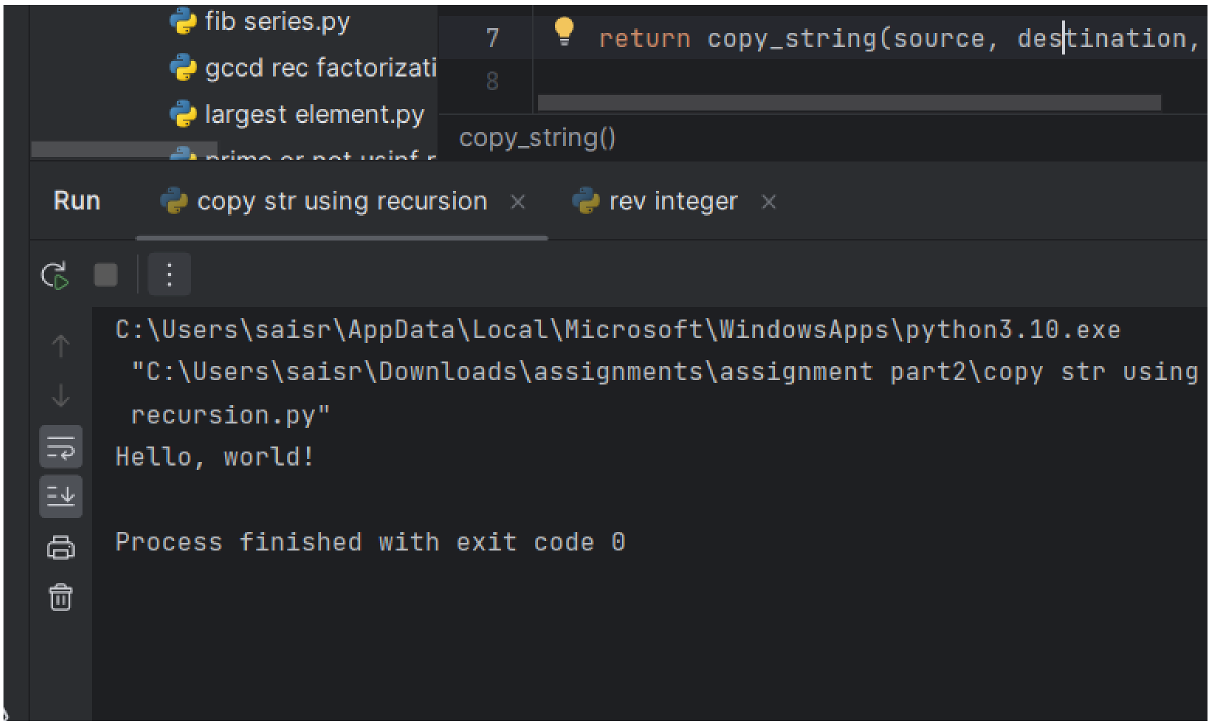
```
def recur_factorial(n):
   if n == 1:
       return n
   else:
       return n*recur_factorial(n-1)

num = 7

# 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))
```



```
C:\Users\saisr\AppData\Local\Microsoft\WindowsApps\python3.10.exe
 "C:\Users\saisr\Downloads\assignments\assignment part2\factorial rec.py"
The factorial of 7 is 5040

Process finished with exit code 0
```

6. Write a program for to copy one string to another using recursion

```
def copy_string(source, destination, index=0):
    if index >= len(source):
        return destination

    destination += source[index]

    return copy_string(source, destination, index + 1)


source_string = "Hello, world!"
destination_string = ""
copied_string = copy_string(source_string, destination_string)
print(copied_string)
```

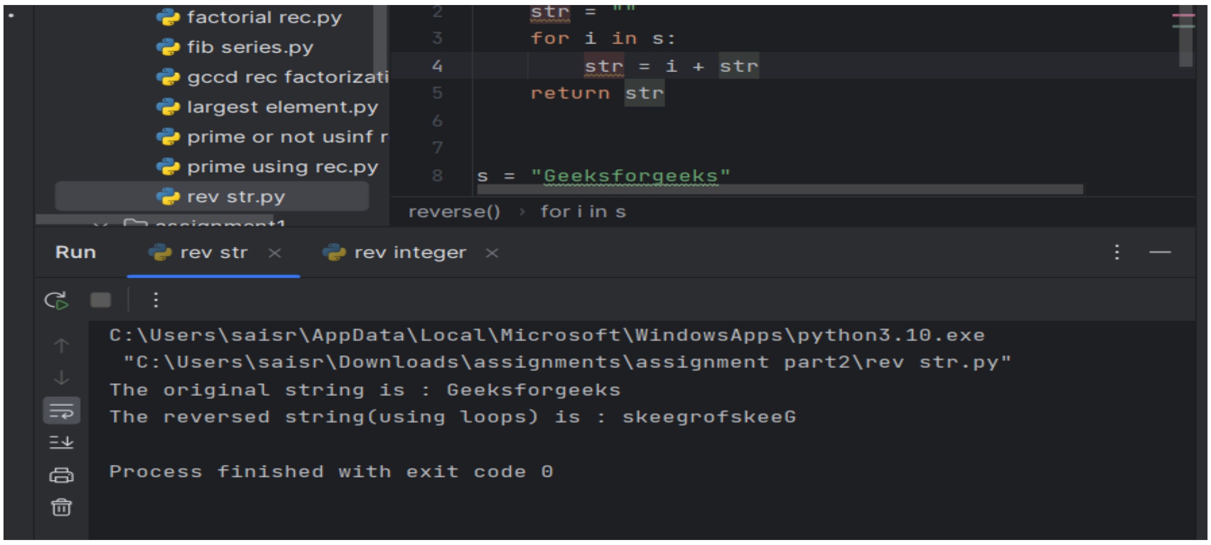


```
    return copy_string(source, destination,
```

```
C:\Users\saisr\AppData\Local\Microsoft\WindowsApps\python3.10.exe
 "C:\Users\saisr\Downloads\assignments\assignment part2\copy str using
 recursion.py"
Hello, world!

Process finished with exit code 0
```

7. Write a program   to print the reverse of a string using recursion

```
def reverse(s):
    str = ""
    for i in s:
        str = i + str
    return str


s = "Geeksforgeeks"

print("The original string is : ", end="")
print(s)

print("The reversed string(using loops) is : ", end="")
print(reverse(s))
```

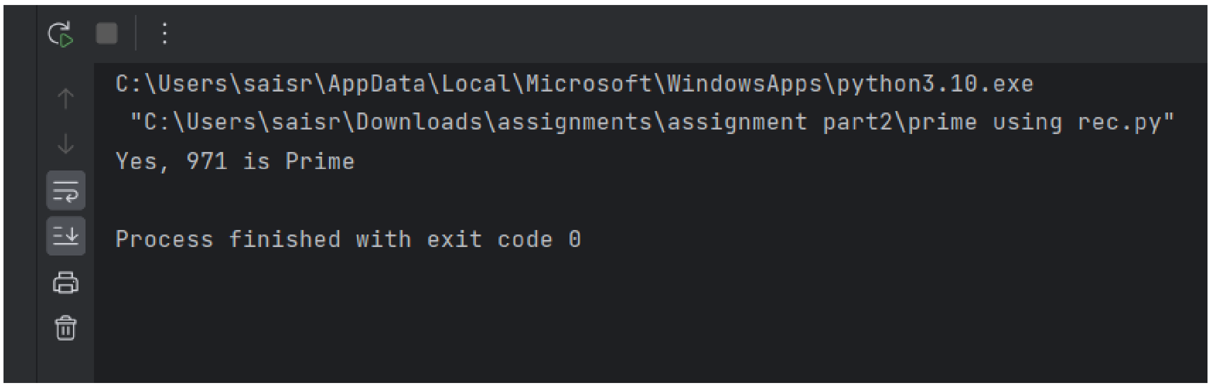


```
    str = ""
    for i in s:
        str = i + str
    return str


s = "Geeksforgeeks"
```

```
C:\Users\saisr\AppData\Local\Microsoft\WindowsApps\python3.10.exe
 "C:\Users\saisr\Downloads\assignments\assignment part2\rev str.py"
The original string is : Geeksforgeeks
The reversed string(using loops) is : skeegrofskeeG

Process finished with exit code 0
```

8. Write a program to generate all the prime numbers using recursion

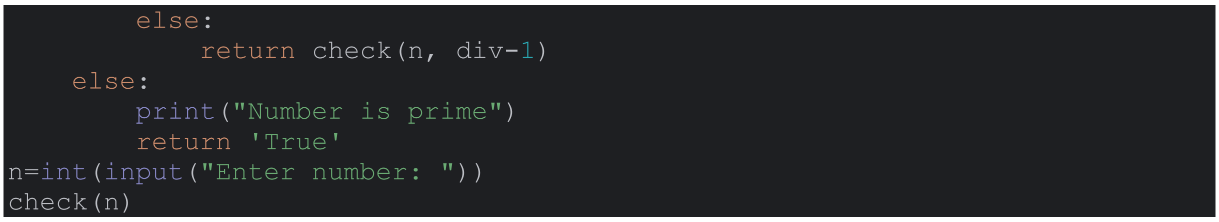
```
def Prime_Number(n, i=2):
    if n == i:
        return True
    elif n % i == 0:
        return False
    return Prime_Number(n, i + 1)


n = 971
if Prime_Number(n):
    print("Yes,", n, "is Prime")
else:
    print("No,", n, "is not a Prime")
```



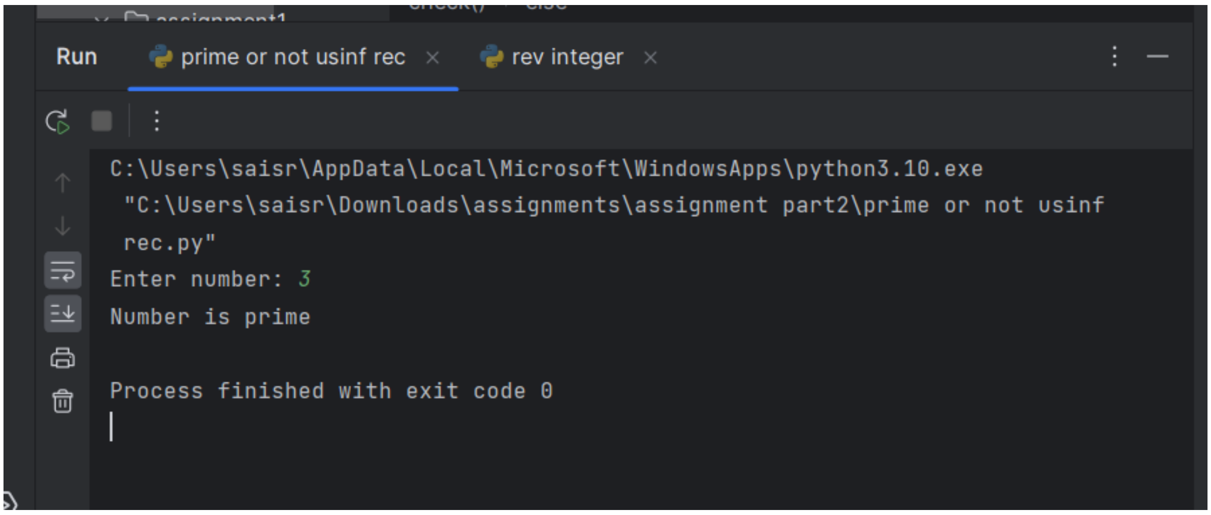
```
C:\Users\saisr\AppData\Local\Microsoft\WindowsApps\python3.10.exe
 "C:\Users\saisr\Downloads\assignments\assignment part2\prime using rec.py"
Yes, 971 is Prime

Process finished with exit code 0
```

9. Write a program to check a number is a prime number or not using recursion.

```
def check(n, div = None):
    if div is None:
        div = n - 1
    while div >= 2:
        if n % div == 0:
            print("Number not prime")
            return False
```

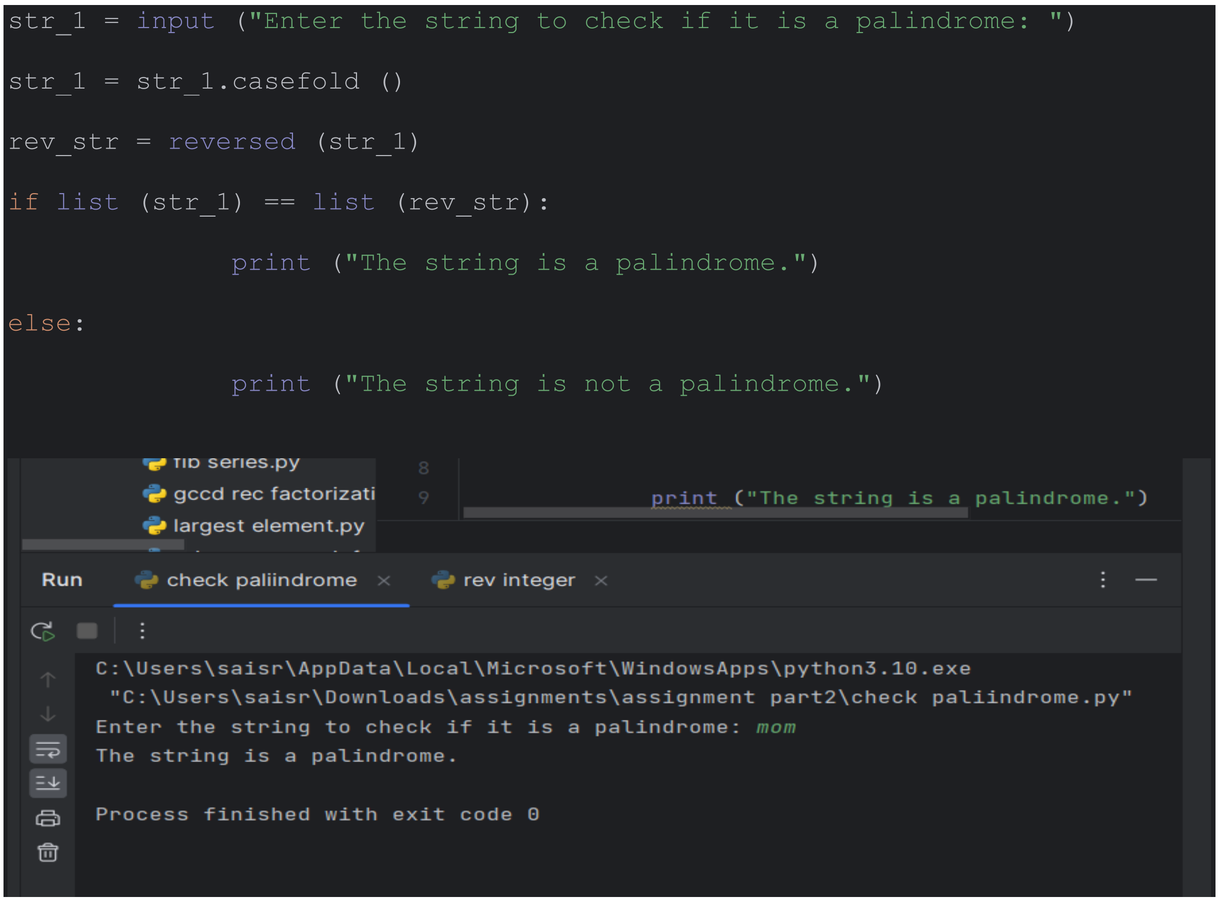


```
        else:
            return check(n, div-1)
    else:
        print("Number is prime")
        return 'True'
n=int(input("Enter number: "))
check(n)
```



```
C:\Users\saisr\AppData\Local\Microsoft\WindowsApps\python3.10.exe
 "C:\Users\saisr\Downloads\assignments\assignment part2\prime or not usinf
 rec.py"
Enter number: 3
Number is prime

Process finished with exit code 0
```

10.Write a program for to check whether a given String is Palindrome or not using recursion



```
str_1 = input ("Enter the string to check if it is a palindrome: ")

str_1 = str_1.casefold ()

rev_str = reversed (str_1)

if list (str_1) == list (rev_str):

              print ("The string is a palindrome.")

else:

              print ("The string is not a palindrome.")
```

```
            print ("The string is a palindrome.")
```

```
C:\Users\saisr\AppData\Local\Microsoft\WindowsApps\python3.10.exe
 "C:\Users\saisr\Downloads\assignments\assignment part2\check paliindrome.py"
Enter the string to check if it is a palindrome: mom
The string is a palindrome.

Process finished with exit code 0
```