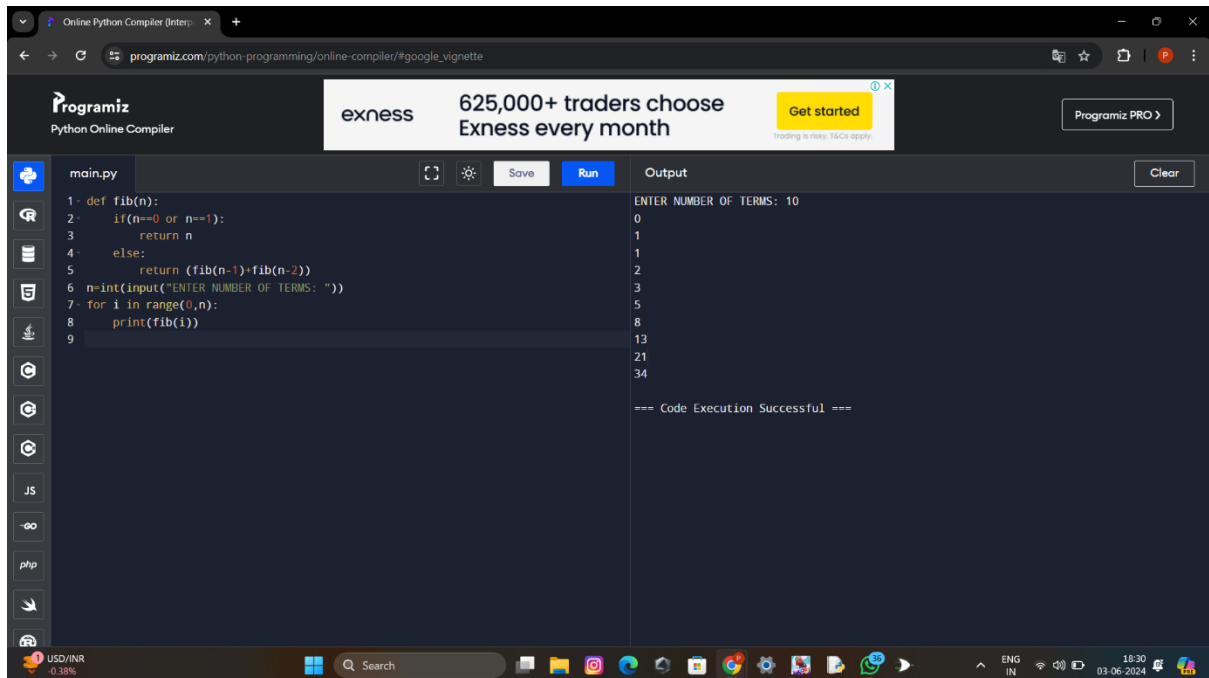


1. Write a program to Print Fibonacci Series using recursion.



The screenshot shows a web browser window with the URL `programiz.com/python-programming/online-compiler/#google_vignette`. The page features a header with the Programiz logo and a banner for Exness. The main area is divided into a code editor and an output window. The code editor contains the following Python code:

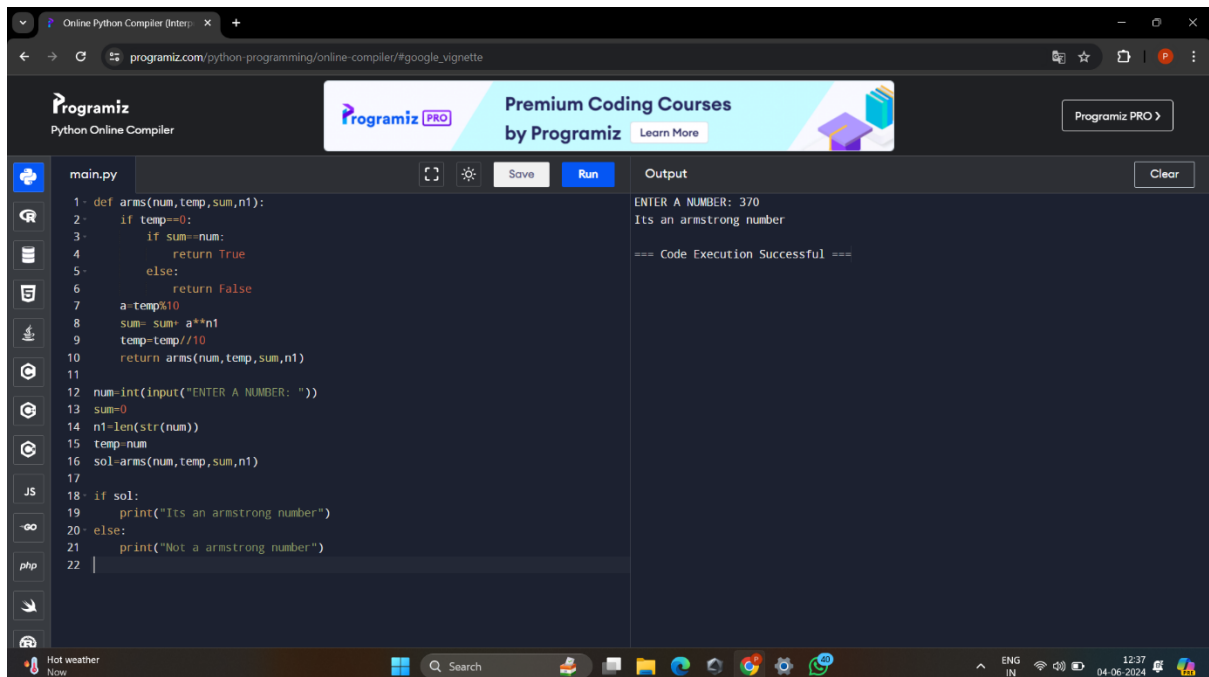
```
1 def fib(n):
2     if(n==0 or n==1):
3         return n
4     else:
5         return (fib(n-1)+fib(n-2))
6 n=int(input("ENTER NUMBER OF TERMS: "))
7 for i in range(0,n):
8     print(fib(i))
9
```

The output window displays the result of the program execution:

```
ENTER NUMBER OF TERMS: 10
0
1
1
2
3
5
8
13
21
34
=== Code Execution Successful ===
```

Time: $O(n^2)$

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



The screenshot shows a web browser window with the URL `programiz.com/python-programming/online-compiler/#google_vignette`. The page features a header with the Programiz logo and a banner for Premium Coding Courses. The main area is divided into a code editor and an output window. The code editor contains the following Python code:

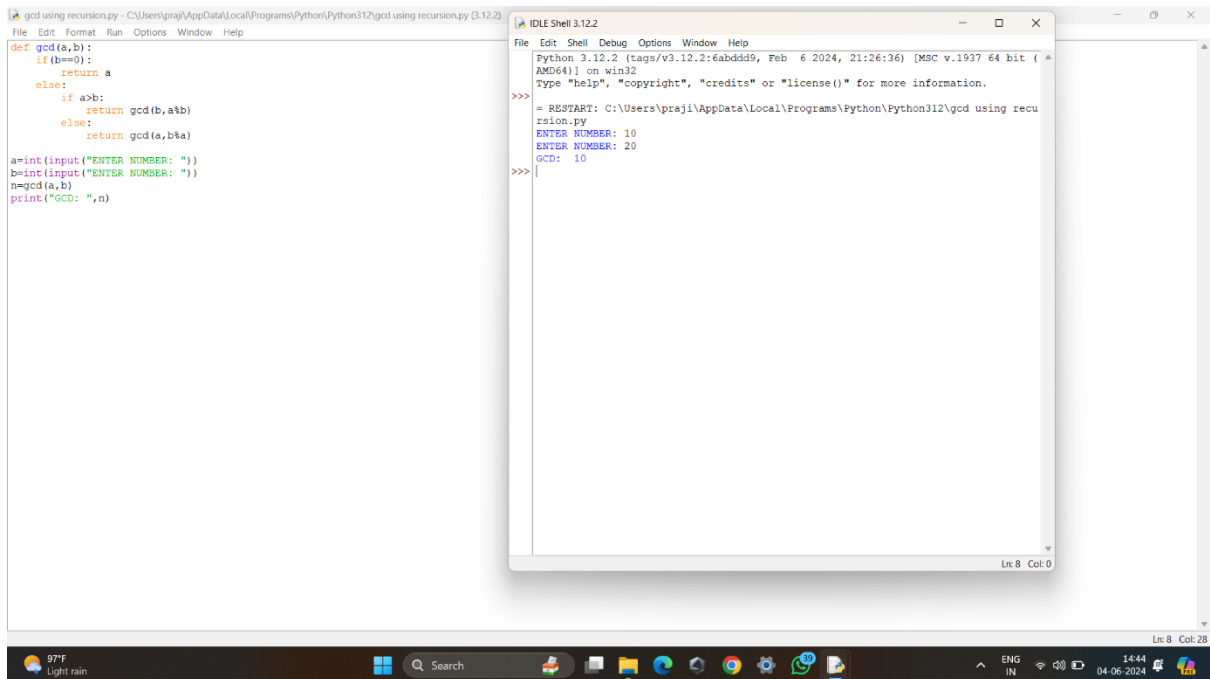
```
1 def arms(num, temp, sum, n1):
2     if temp==0:
3         if sum==num:
4             return True
5         else:
6             return False
7     a=temp%10
8     sum= sum+ a**n1
9     temp=temp//10
10    return arms(num, temp, sum, n1)
11
12 num=int(input("ENTER A NUMBER: "))
13 sum=0
14 n1=len(str(num))
15 temp=num
16 sol=arms(num, temp, sum, n1)
17
18 if sol:
19     print("Its an armstrong number")
20 else:
21     print("Not a armstrong number")
22
```

The output window displays the result of the program execution:

```
ENTER A NUMBER: 370
Its an armstrong number
=== Code Execution Successful ===
```

Time: $O(n)$

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



The screenshot shows a Python IDE with two windows. The main window displays a Python script for finding the GCD of two numbers using recursion. The script defines a function `gcd(a,b)` that returns `a` if `b` is 0, otherwise it returns `gcd(b, a%b)`. It prompts the user to enter two numbers and prints the GCD. The second window shows the execution of the program, where the user enters 10 and 20, and the output is GCD: 10.

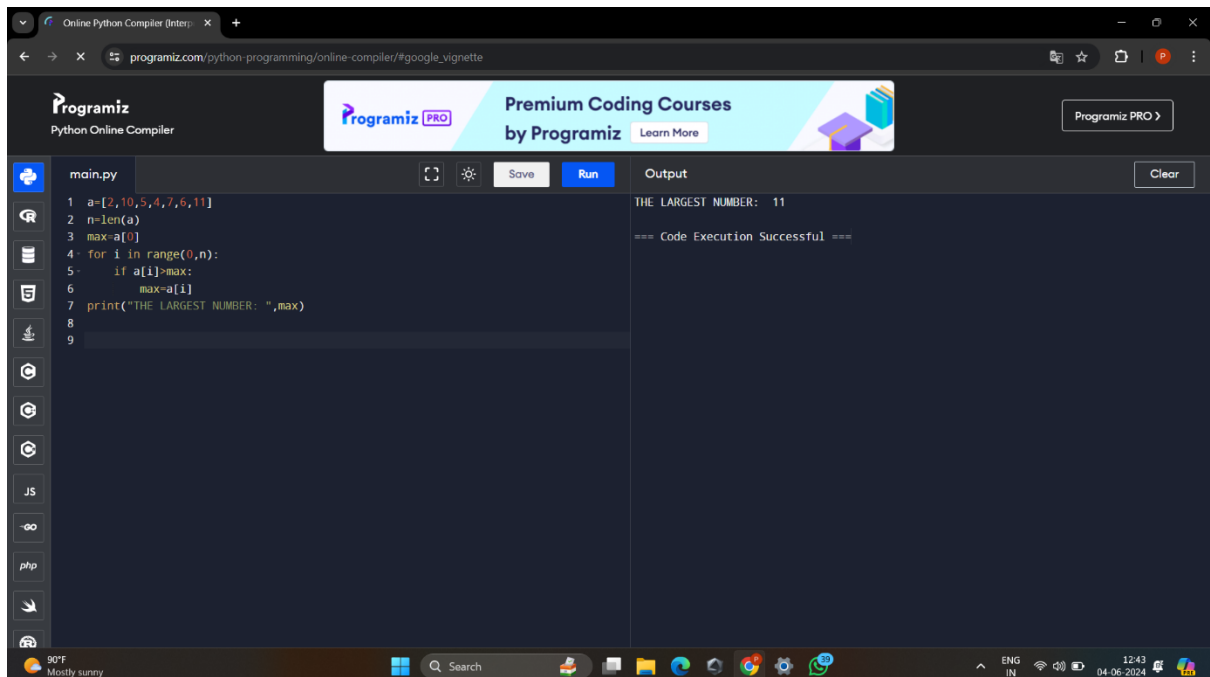
```
def gcd(a,b):
    if (b==0):
        return a
    else:
        if a>b:
            return gcd(b,a%b)
        else:
            return gcd(a,b%a)

a=int(input("ENTER NUMBER: "))
b=int(input("ENTER NUMBER: "))
n=gcd(a,b)
print("GCD: ",n)
```

```
Python 3.12.2 (tags/v3.12.2:6abdd9, Feb 6 2024, 21:26:36) [MSC v.1937 64 bit (AMD64)] on win32
Type "help", "copyright", "credits" or "license()" for more information.
>>>
= RESTART: C:\Users\praji\AppData\Local\Programs\Python\Python312\gcd using recursion.py
ENTER NUMBER: 10
ENTER NUMBER: 20
GCD: 10
>>>
```

Time: $O(n)$

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



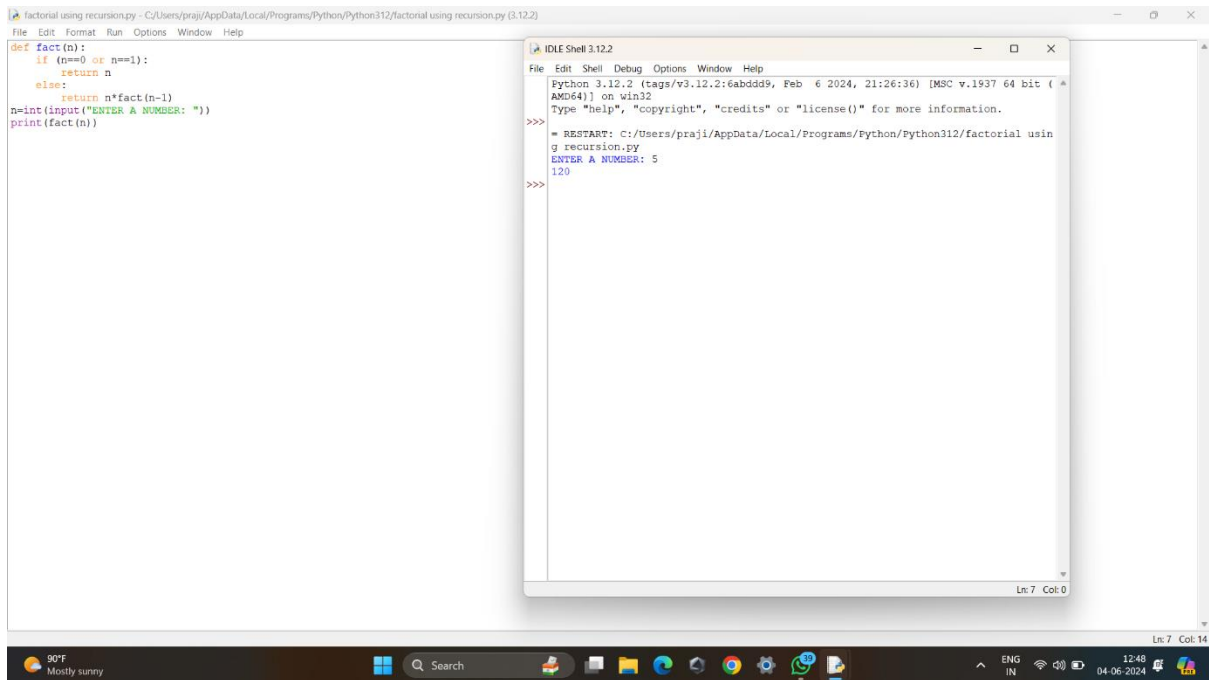
The screenshot shows an online Python compiler interface. The code in the editor finds the largest element in an array `a` by iterating through it and comparing each element to the current maximum. The output shows the largest number is 11.

```
main.py
1 a=[2,10,5,4,7,6,11]
2 n=len(a)
3 max=a[0]
4 for i in range(0,n):
5     if a[i]>max:
6         max=a[i]
7 print("THE LARGEST NUMBER: ",max)
8
9
```

```
THE LARGEST NUMBER: 11
=== Code Execution Successful ===
```

Time: $O(n)$

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



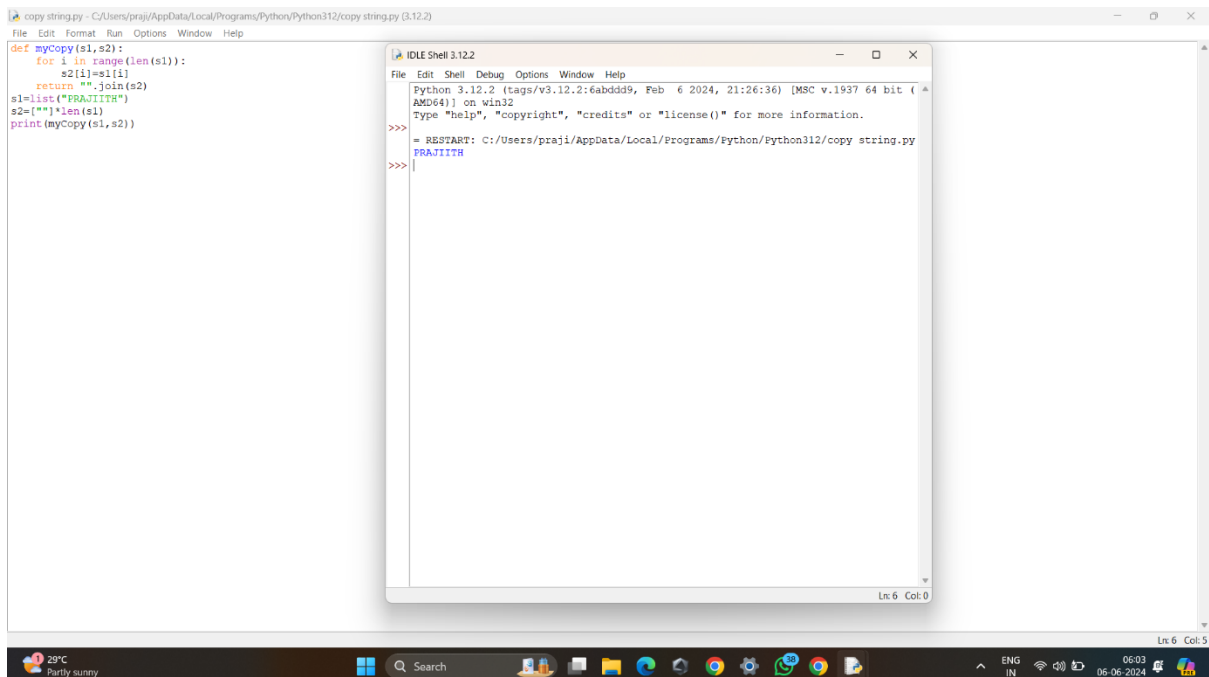
The screenshot shows a Python IDE with two windows. The main window displays a file named 'factorial using recursion.py' with the following code:

```
def fact(n):  
    if (n==0 or n==1):  
        return n  
    else:  
        return n*fact(n-1)  
n=int(input("ENTER A NUMBER: "))  
print(fact(n))
```

The 'IDLE Shell 3.12.2' window shows the execution of the program. It displays the prompt 'ENTER A NUMBER: 5' and the output '120'.

Time: $O(n)$

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

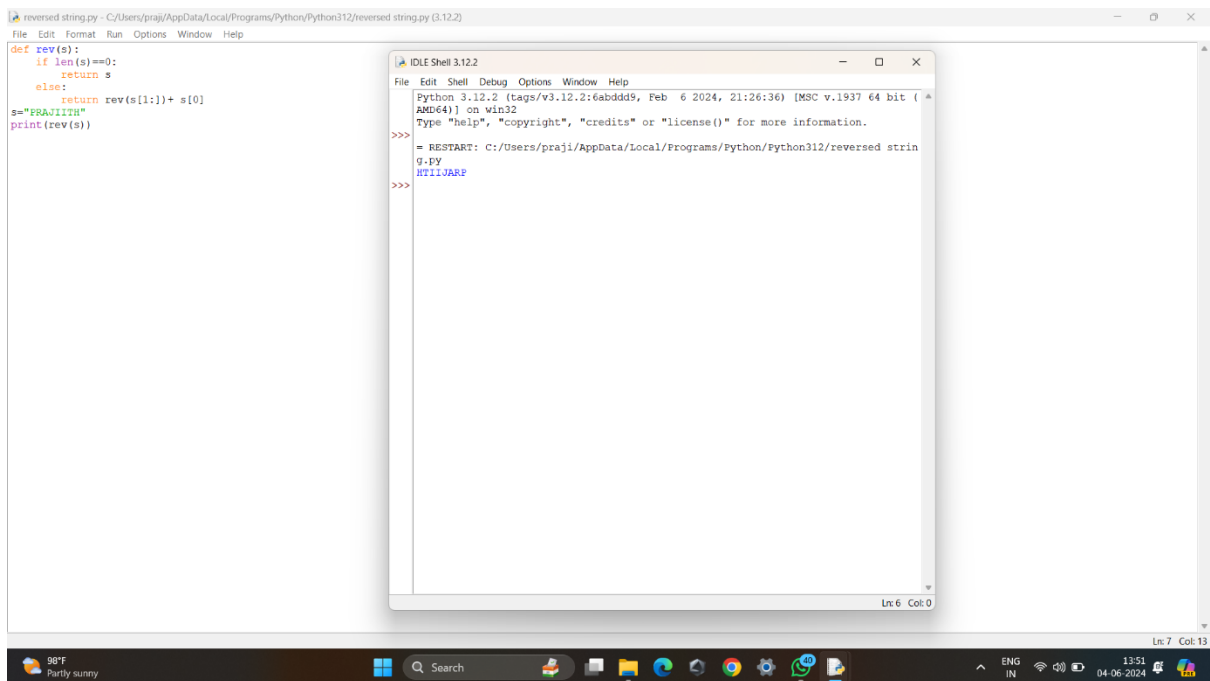


The screenshot shows a Python IDE with two windows. The main window displays a file named 'copy string.py' with the following code:

```
def myCopy(s1,s2):  
    for i in range(len(s1)):  
        s2[i]=s1[i]  
    return "".join(s2)  
s1=list("PRAJITH")  
s2=[""]*len(s1)  
print(myCopy(s1,s2))
```

The 'IDLE Shell 3.12.2' window shows the execution of the program. It displays the prompt 'PRAJITH' and the output 'PRAJITH'.

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



The screenshot shows an IDE window titled 'reversed string.py' with the following Python code:

```
def rev(s):
    if len(s)==0:
        return s
    else:
        return rev(s[1:])+ s[0]
s="PRAJITH"
print(rev(s))
```

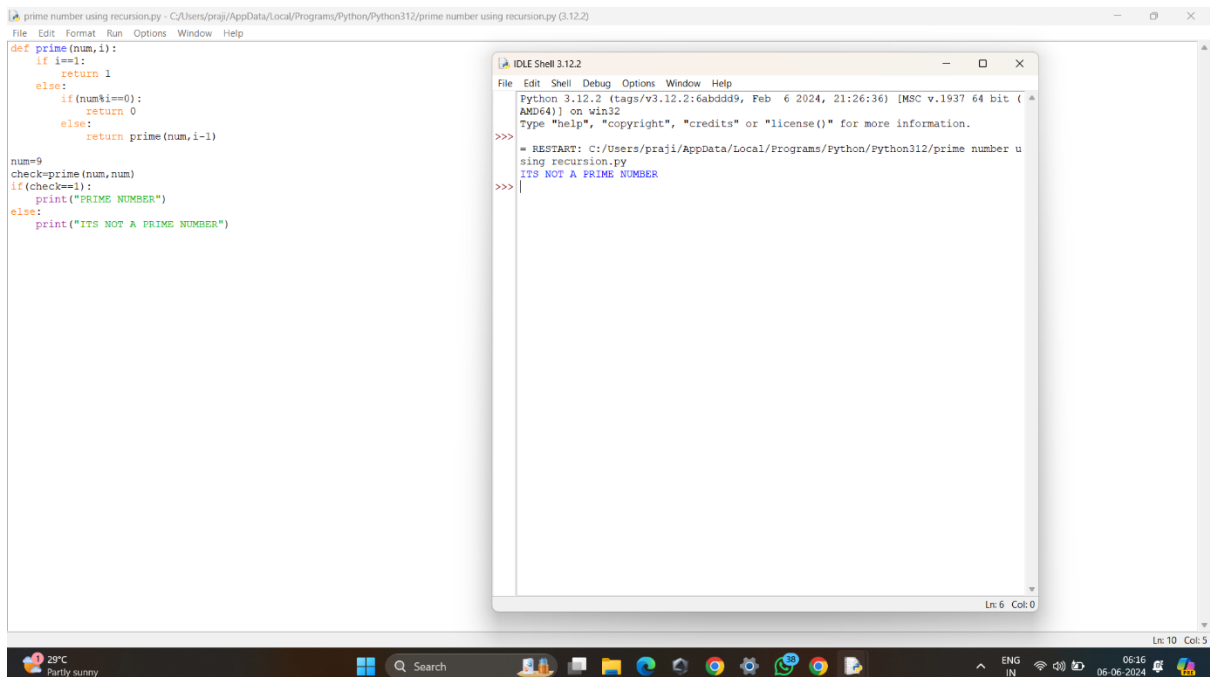
An 'IDLE Shell 3.12.2' window is open, displaying the output of the program:

```
>>>
= RESTART: C:/Users/praji/AppData/Local/Programs/Python/Python312/reversed string.py
HTIIOAREP
>>>
```

The Windows taskbar at the bottom shows the date as 04-06-2024 and the time as 13:51.

Time: $O(n)$

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



The screenshot shows an IDE window titled 'prime number using recursion.py' with the following Python code:

```
def prime(num,i):
    if i==1:
        return 1
    else:
        if (num%i==0):
            return 0
        else:
            return prime(num,i-1)

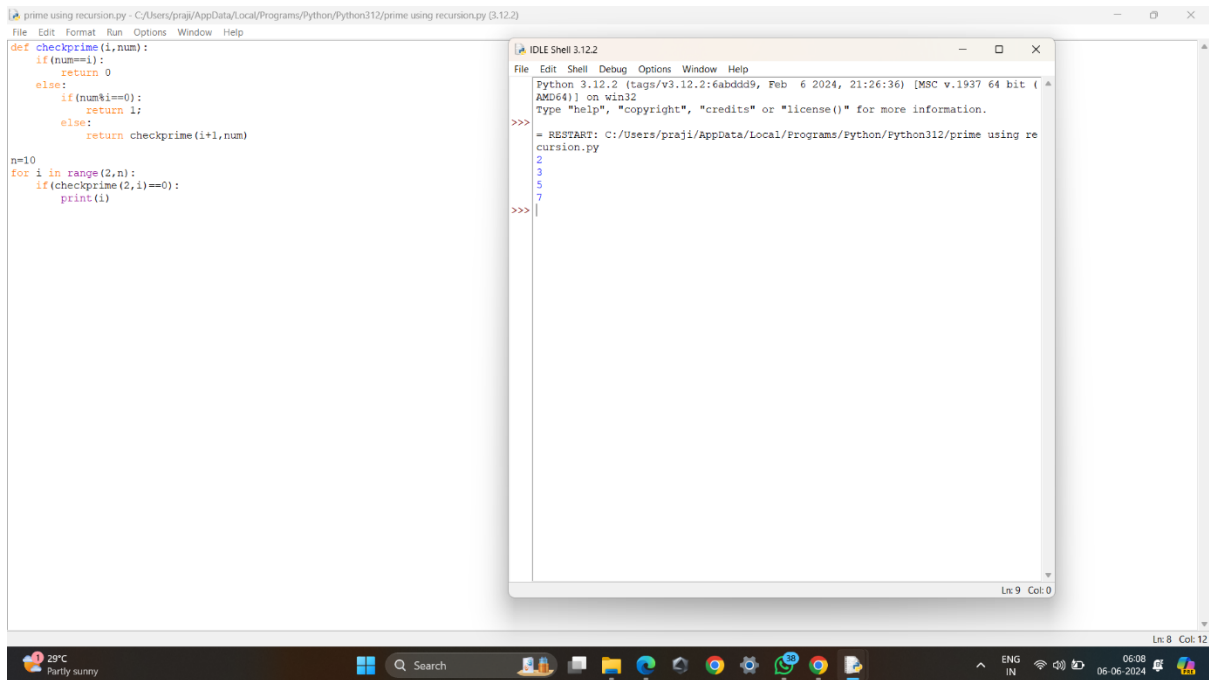
num=9
check=prime(num,num)
if (check==1):
    print("PRIME NUMBER")
else:
    print("ITS NOT A PRIME NUMBER")
```

An 'IDLE Shell 3.12.2' window is open, displaying the output of the program:

```
>>>
= RESTART: C:/Users/praji/AppData/Local/Programs/Python/Python312/prime number using recursion.py
ITS NOT A PRIME NUMBER
>>>
```

The Windows taskbar at the bottom shows the date as 06-06-2024 and the time as 06:16.

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

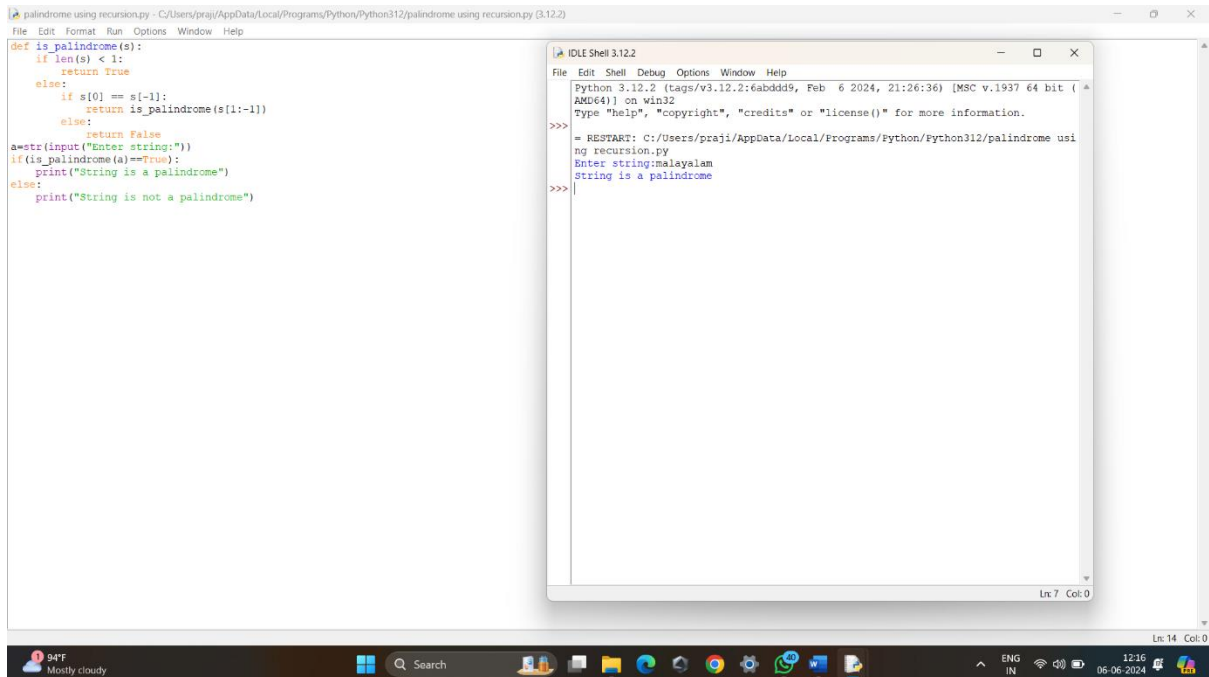


```
prime using recursion.py - C:/Users/praji/AppData/Local/Programs/Python/Python312/prime using recursion.py (3.12.2)
File Edit Format Run Options Window Help
def checkprime(i,num):
    if(num==i):
        return 0
    else:
        if(num%i==0):
            return 1;
        else:
            return checkprime(i+1,num)
n=10
for i in range(2,n):
    if(checkprime(2,i)==0):
        print(i)

IDLE Shell 3.12.2
Python 3.12.2 (tags/v3.12.2:6abddd9, Feb 6 2024, 21:26:36) [MSC v.1937 64 bit (AMD64)] on win32
Type "help", "copyright", "credits" or "license()" for more information.
>>>
= RESTART: C:/Users/praji/AppData/Local/Programs/Python/Python312/prime using recursion.py
2
3
5
7
>>>
```

Time:O(n)

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



```
palindrome using recursion.py - C:/Users/praji/AppData/Local/Programs/Python/Python312/palindrome using recursion.py (3.12.2)
File Edit Format Run Options Window Help
def is_palindrome(s):
    if len(s) < 1:
        return True
    else:
        if s[0] == s[-1]:
            return is_palindrome(s[1:-1])
        else:
            return False
a= str(input("Enter string:"))
if is_palindrome(a)==True:
    print("String is a palindrome")
else:
    print("String is not a palindrome")

IDLE Shell 3.12.2
Python 3.12.2 (tags/v3.12.2:6abddd9, Feb 6 2024, 21:26:36) [MSC v.1937 64 bit (AMD64)] on win32
Type "help", "copyright", "credits" or "license()" for more information.
>>>
= RESTART: C:/Users/praji/AppData/Local/Programs/Python/Python312/palindrome using recursion.py
Enter string:malayalam
String is a palindrome
>>>
```

Time:O(n)

