

# Cryptography & Network Security

PRN - 2019BTECS00026

Name - Niraja Vasudev Kulkarni

Batch - B1

## Assignment - 9

**Title:** Prime Factorization

**Aim:** To Demonstrate Prime Factorization

**Theory:**

RSA Laboratories states that: for each RSA number  $n$ , there exists prime numbers  $p$  and  $q$  such that

$$n = p \times q.$$

The problem is to find these two primes, given only  $n$ .

**Code:**

```
from sympy.ntheory import factorint
import math

def factors_int(num):
    poss_p = math.floor(math.sqrt(num))

    if poss_p % 2 == 0:
        poss_p += 1

    while poss_p < num:
```

```

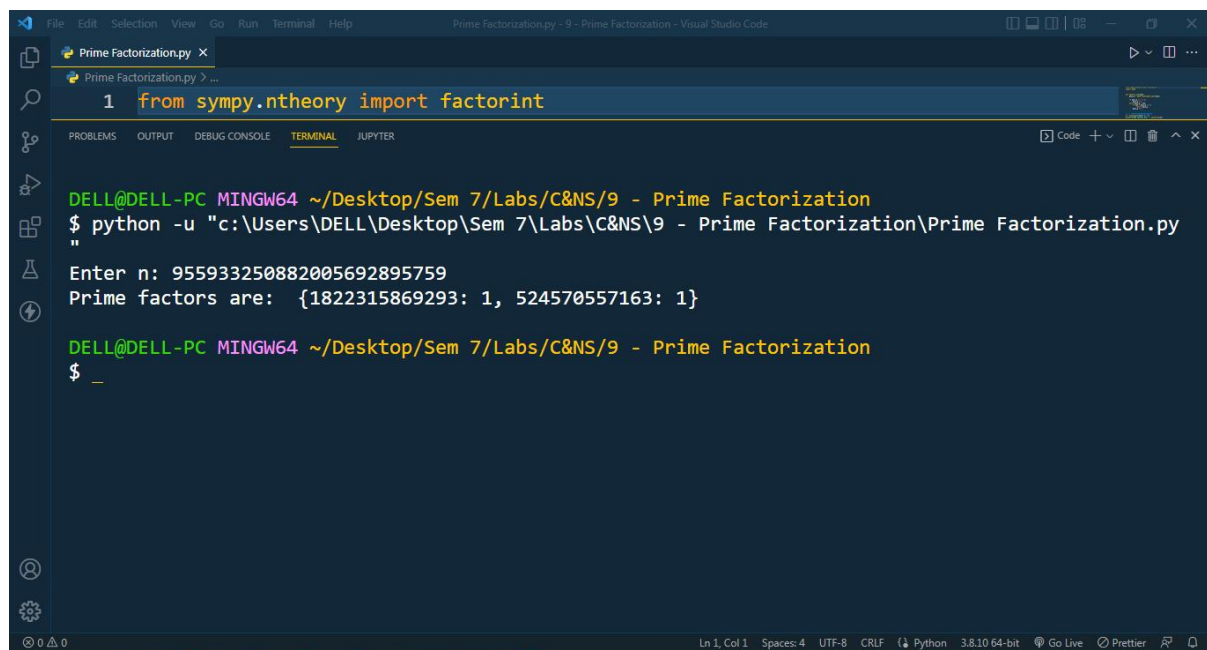
if num % poss_p == 0:
    return poss_p

poss_p += 2

# n = 955933250882005692895759
n = int(input("Enter n: "))
print(factorint(n))

```

### Output:



The screenshot shows a Visual Studio Code editor with a file named 'Prime Factorization.py' open. The code in the editor is:
 

```
1 from sympy.ntheory import factorint
```

 Below the editor, the 'TERMINAL' tab is active, displaying the following output:
 

```

DELL@DELL-PC MINGW64 ~/Desktop/Sem 7/Labs/C&NS/9 - Prime Factorization
$ python -u "c:\Users\DELL\Desktop\Sem 7\Labs\C&NS\9 - Prime Factorization\Prime Factorization.py"
Enter n: 955933250882005692895759
Prime factors are: {1822315869293: 1, 524570557163: 1}
DELL@DELL-PC MINGW64 ~/Desktop/Sem 7/Labs/C&NS/9 - Prime Factorization
$ _
  
```

 The status bar at the bottom indicates the file is at 'Ln 1, Col 1' with 'Spaces: 4', 'UTF-8' encoding, 'CRLF' line endings, and is using 'Python 3.8.10 64-bit' interpreter.

### Conclusion:

The RSA Factoring Challenge was a challenge put forward by RSA Laboratories to encourage research into computational number theory and the practical difficulty of factoring large integers and cracking RSA keys used in cryptography. They published a list of semiprimes (numbers with exactly two prime factors) known as the RSA numbers, with a cash prize for the successful factorization of some of them.