

Assignment: 1 Python Programming – Number Theory Functions

Assignment google colab Url - <https://colab.research.google.com/drive/1pRCMOj3U-oXU2EP2LrggKirH5pTqWzIN?usp=sharing>

Task 1: Check if a Number is Prime

Write a Python function to determine whether a given integer is a prime number.

Function Specifications:

o Input: An integer.

o Output: True if the number is prime, otherwise False.

```
#Task 1: Check if a Number is Prime
try:
    n = int(input("Enter a number to check whether it is prime or not: ")) #input number wanted to check

    def is_prime(n):
        if n <= 1: #numbers below 0 doesnt need to consider
            return False
        elif n == 2:
            return True
        elif n % 2 == 0: #if a number is divisible by 2 then it is not a prime
            return False
        else:
            i = 3
            while i * i <= n: # Only need to check until the square root of n
                if n % i == 0:
                    return False
                i += 2 #increment i to avoid even numbers
            return True

    #output
    if is_prime(n):
        print("True")
    else:
        print("False")

except ValueError:
    print("Invalid input. Please enter a valid integer.")
```

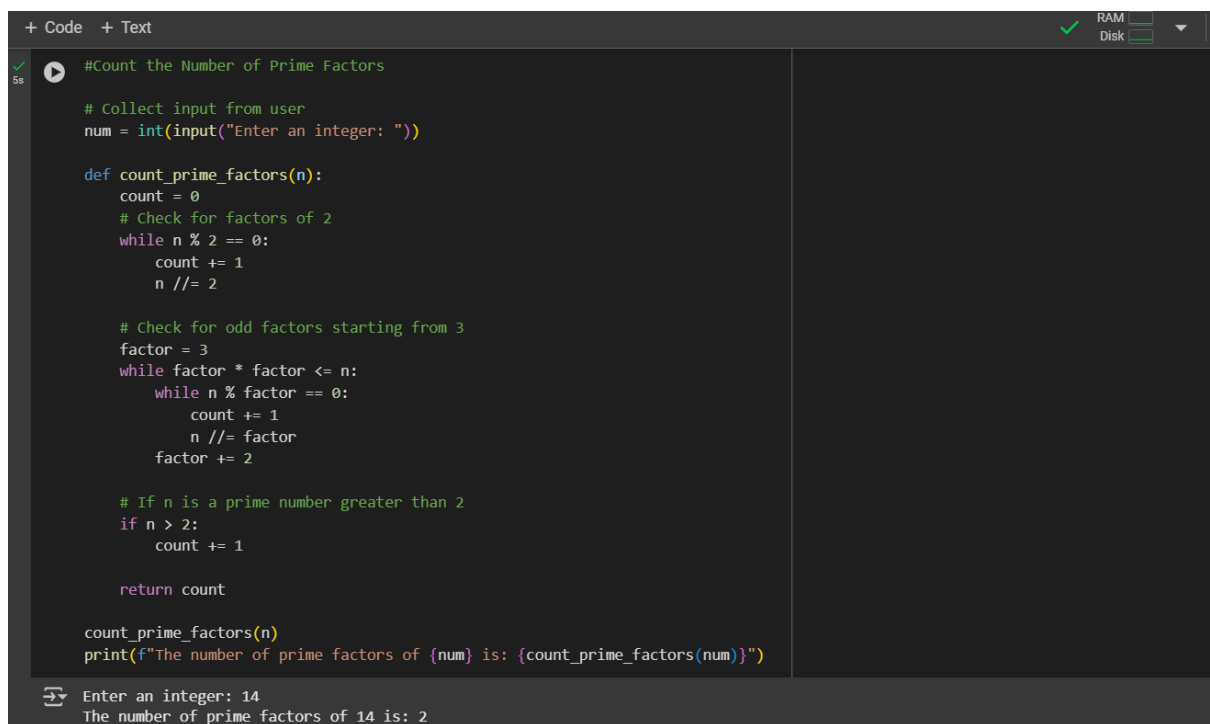
Enter a number to check whether it is prime or not: 17
True

Task 2: Count the Number of Prime Factors

Write a Python function to count the number of prime factors of a given integer.

Function Specifications:

- **Input:** An integer.
- **Output:** The count of prime factors of the number.



The screenshot shows a Python IDE with a dark theme. The top bar has tabs for '+ Code' and '+ Text', and a status bar on the right showing 'RAM' and 'Disk' usage. The main editor area contains the following Python code:

```
#Count the Number of Prime Factors

# Collect input from user
num = int(input("Enter an integer: "))

def count_prime_factors(n):
    count = 0
    # Check for factors of 2
    while n % 2 == 0:
        count += 1
        n //= 2

    # Check for odd factors starting from 3
    factor = 3
    while factor * factor <= n:
        while n % factor == 0:
            count += 1
            n //= factor
        factor += 2

    # If n is a prime number greater than 2
    if n > 2:
        count += 1

    return count

count_prime_factors(num)
print(f"The number of prime factors of {num} is: {count_prime_factors(num)}")
```

Below the code editor, the input and output are shown:


```
Enter an integer: 14
The number of prime factors of 14 is: 2
```

Task 3: Implement the Euclidean Algorithm to Find GCD

Implement the Euclidean Algorithm in Python to compute the greatest common divisor (GCD) of two integers.


Function Specifications:

- Input: Two integers.
- Output: The GCD of the two integers.

```
 #GCD using Euclidean Algorithm
def gcd(a, b):
    while b:
        a, b = b, a % b
    return a

# Taking input from user
num1 = int(input("Enter first number: "))
num2 = int(input("Enter second number: "))

# Finding and printing GCD
print("GCD:", gcd(num1, num2))
```

```
 Enter first number: 10
Enter second number: 20
GCD: 10
```

Task 4: Find the Least Common Multiple (LCM)

Write a Python function to calculate the least common multiple (LCM) of two integers using GCD.

Function Specifications:

- **Input:** Two integers.
- **Output:** The LCM of the two integers.

```
GCD: 10

17s #LCM using GCD
def gcd(a, b):
    while b:
        a, b = b, a % b
    return a

def lcm(a, b):
    return abs(a * b) // gcd(a, b)

# Taking input from user
num1 = int(input("Enter first number: "))
num2 = int(input("Enter second number: "))

# Finding and printing LCM
print("LCM:", lcm(num1, num2))

Enter first number: 40
Enter second number: 60
LCM: 120
```

SUBMITTED BY

GOVIND V KARTHA - 241026