File Edit View Insert Runtime Tools Help

Commands + Code + Text

[2] 0s
```
def greet():
    print("Hello, python!")

greet()
```

```
Hello, python!
```

Generate | a slider using jupyter widgets | Close

[4] 0s
```
def sum_of_two(a, b):
    return a + b

print("Sum:", sum_of_two(5, 3))
```

```
Sum: 8
```

[5] 0s
```
def factorial(n):
    return 1 if n == 0 else n * factorial(n - 1)

print("Factorial:", factorial(5))
```

```
Factorial: 120
```

File Edit View Insert Runtime Tools Help

Commands + Code + Text

[7] 3s
```
def is_prime(n):
    if n < 2:
        return False
    for i in range(2, int(n ** 0.5) + 1):
        if n % i == 0:
            return False
    return True

num = int(input("Enter a number: "))
print("Prime" if is_prime(num) else "Not Prime")
```

```
Enter a number: 26
Not Prime
```

[11] 3s
```
#fibonnaci series generator
def fibonacci(n):
    a, b = 0, 1
    for _ in range(n):
        print(a, end=" ")
        a, b = b, a + b

num = int(input("Enter the number of terms: "))
fibonacci(num)
```

```
Enter the number of terms: 46
0 1 1 2 3 5 8 13 21 34 55 89 144 233 377 610 987 1597 2584 4181 6765 10946 17711 28657 46368 75025 121393 196418
```

Commands | + Code + Text

Generated code may be subject to a license | Abhishek991926/python_program

```
#largest of three numbers
def largest_of_three(a, b, c):
    return max(a, b, c)

num1 = int(input("Enter the first number: "))
num2 = int(input("Enter the second number: "))
num3 = int(input("Enter the third number: "))

print("Largest:", largest_of_three(num1, num2, num3))
```

```
Enter the first number: 25
Enter the second number: 23
Enter the third number: 12
Largest: 25
```

```
[13] #Pallindrome checker
def is_palindrome(s):
    return s == s[::-1]

word = input("Enter a word: ")
print("Pallindrome" if is_palindrome(word) else "Not Pallindrome")
```

```
Enter a word: 23
Not Pallindrome
```

```
[29] #Armstrong number checker
def is_armstrong(n):
    num_str = str(n)
    power = len(num_str)
    return n == sum(int(digit) ** power for digit in num_str)

num = int(input("Enter a number: "))
print("Armstrong Number" if is_armstrong(num) else "Not an Armstrong Number")
```

```
Enter a number: 78
Not an Armstrong Number
```

```
[30] #Power function
def power(base, exponent):
    return base ** exponent

base = int(input("Enter the base: "))
exponent = int(input("Enter the exponent: "))

print("Result:", power(base, exponent))
```

```
Enter the base: 22
Enter the exponent: 55
Result: 681157346867706867423648879192255614042355433059946411344464814211260743 68
```

File Edit View Insert Runtime Tools Help

Commands + Code + Text

```
[34] Enter second number: 12
     Enter operator (+, -, *, /): 65
     Result: Invalid Operator
```

```
[38] #even or odd function
     def even_or_odd(n):
         return "Even" if n % 2 == 0 else "Odd"

     num = int(input("Enter a number: "))
     print(even_or_odd(num))
```

```
Enter a number: 85
Odd
```

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

num1 = int(input("Enter first number: "))
num2 = int(input("Enter second number: "))

print("GCD:", gcd(num1, num2))
```

```
Enter first number: 21
Enter second number: 64
GCD: 1
```

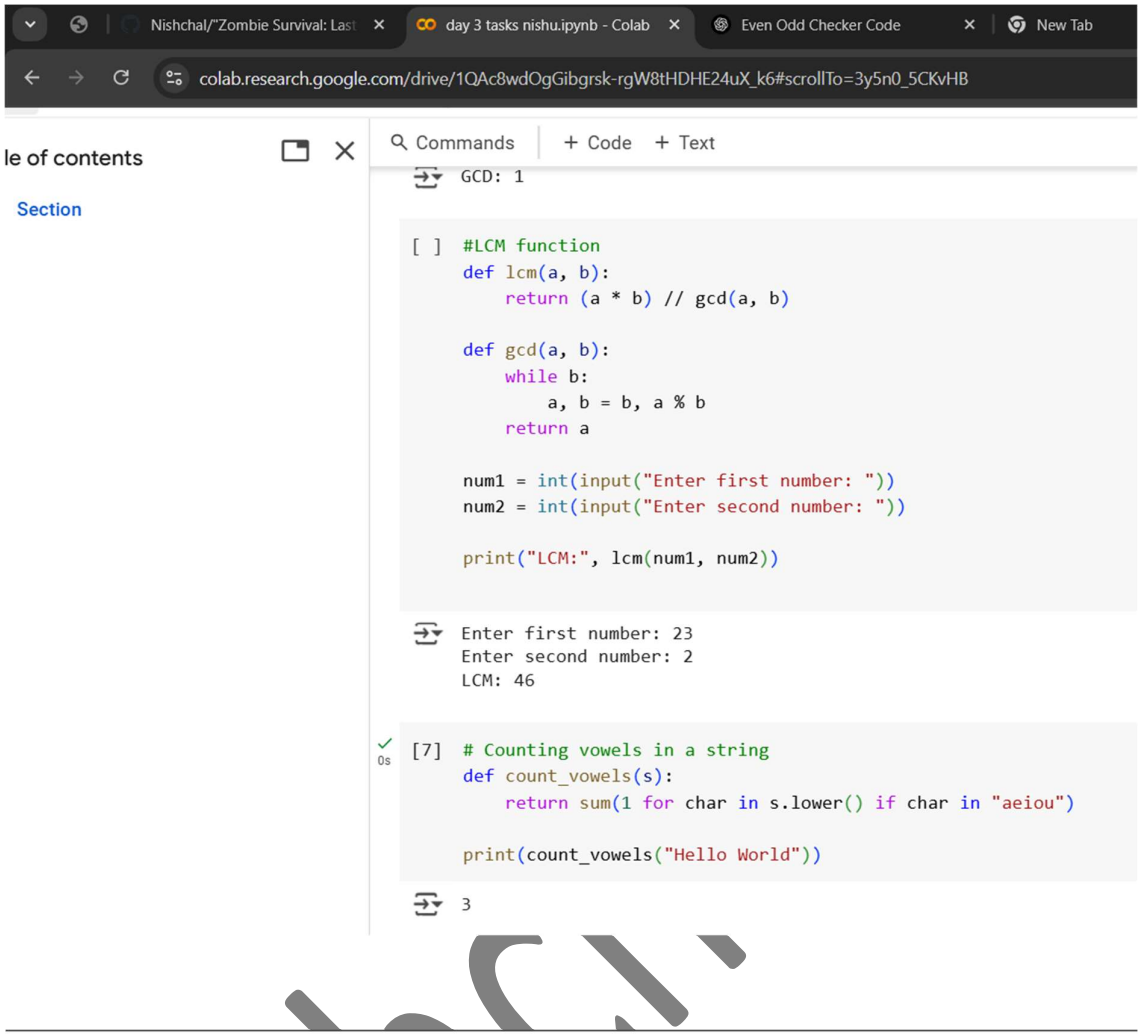

10s

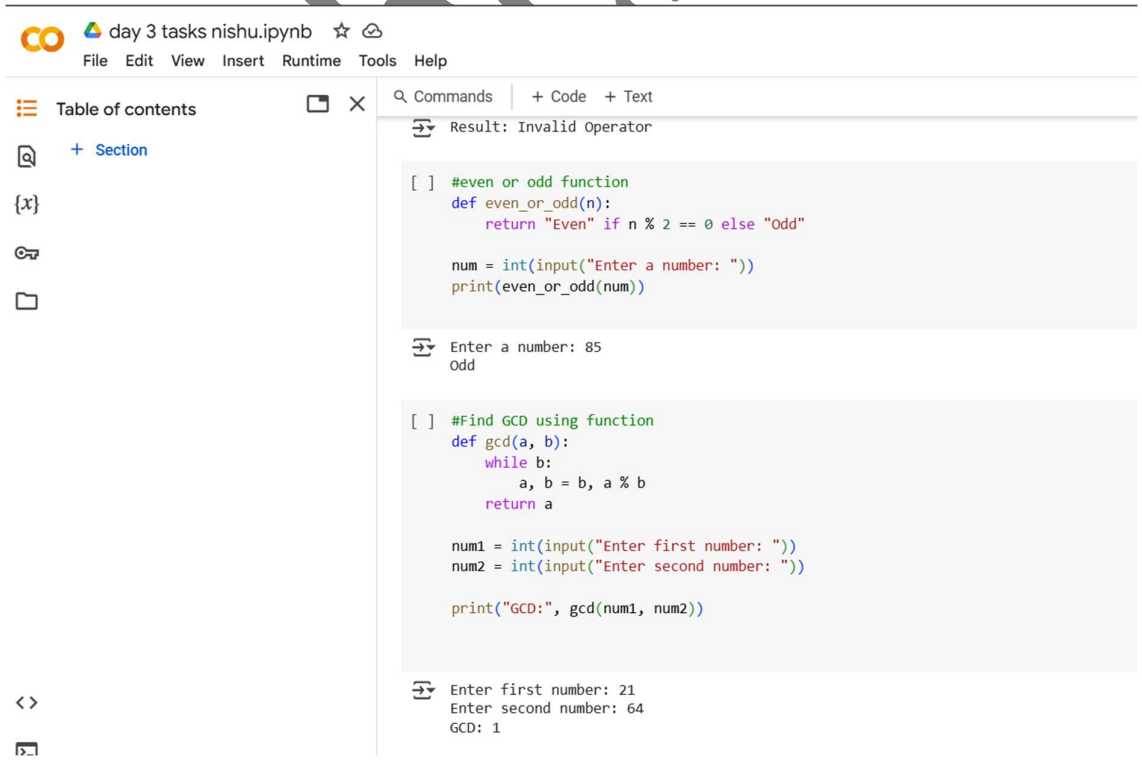

```
#simple calculator
def calculator(a, b, op):
    if op == '+':
        return a + b
    elif op == '-':
        return a - b
    elif op == '*':
        return a * b
    elif op == '/':
        return a / b if b != 0 else "Error: Division by zero"
    else:
        return "Invalid Operator"

num1 = float(input("Enter first number: "))
num2 = float(input("Enter second number: "))
operator = input("Enter operator (+, -, *, /): ")

print("Result:", calculator(num1, num2, operator))
```

```
Enter first number: 23
Enter second number: 12
Enter operator (+, -, *, /): 65
Result: Invalid Operator
```


Nishchal/"Zombie Survival: Last
day 3 tasks nishu.ipynb - Colab
Even Odd Checker Code
New Tab
colab.research.google.com/drive/1QAc8wdOgGibgrsk-rgW8tHDHE24uX_k6#scrollTo=3y5n0_5CKvHB
le of contents
Section
Commands
+ Code
+ Text
GCD: 1
[ ] #LCM function
def lcm(a, b):
return (a * b) // gcd(a, b)
def gcd(a, b):
while b:
a, b = b, a % b
return a
num1 = int(input("Enter first number: "))
num2 = int(input("Enter second number: "))
print("LCM:", lcm(num1, num2))
Enter first number: 23
Enter second number: 2
LCM: 46
0s
[7] # Counting vowels in a string
def count_vowels(s):
return sum(1 for char in s.lower() if char in "aeiou")
print(count_vowels("Hello World"))
3


day 3 tasks nishu.ipynb
File Edit View Insert Runtime Tools Help
Table of contents
+ Section
Commands
+ Code
+ Text
Result: Invalid Operator
[ ] #even or odd function
def even_or_odd(n):
return "Even" if n % 2 == 0 else "Odd"
num = int(input("Enter a number: "))
print(even_or_odd(num))
Enter a number: 85
Odd
[ ] #Find GCD using function
def gcd(a, b):
while b:
a, b = b, a % b
return a
num1 = int(input("Enter first number: "))
num2 = int(input("Enter second number: "))
print("GCD:", gcd(num1, num2))
Enter first number: 21
Enter second number: 64
GCD: 1

colab.research.google.com/drive/1QAc8wdOgGibgrsk-rgW8tHDHE24uX_k6#scrollTo=3y5n0_5CKvHB

le of contents

Section

Commands | + Code | + Text

```
[8]  # Sum of digits function
     def sum_of_digits(n):
         return sum(int(digit) for digit in str(n))

     print(sum_of_digits(1234))
```

```
10
```

Generate | create a dataframe with 2 columns and 10 rows

```
[9]  # Default arguments function
     def greet(name="Guest"):
         return f"Hello, {name}!"

     print(greet())
     print(greet("Alice"))
```

```
Hello, Guest!
Hello, Alice!
```

```
[10] # Using *args in function
     def sum_all(*args):
         return sum(args)

     print(sum_all(1, 2, 3, 4, 5))
```

```
15
```

Nishchal/"Zombie Survival: Last | day 3 tasks nishu.ipynb - Colab | Even Odd Checker Code | New Tab

colab.research.google.com/drive/1QAc8wdOgGibgrsk-rgW8tHDHE24uX_k6#scrollTo=3y5n0_5CKvHB

le of contents

Section

Commands | + Code | + Text

```
[11] # Using **kwargs in function
     def user_info(**kwargs):
         return kwargs

     print(user_info(name="Alice", age=25, city="New York"))
```

```
{'name': 'Alice', 'age': 25, 'city': 'New York'}
```

```
[12] # Using the math module
     import math
     print(math.sqrt(25))
     print(math.factorial(5))
```

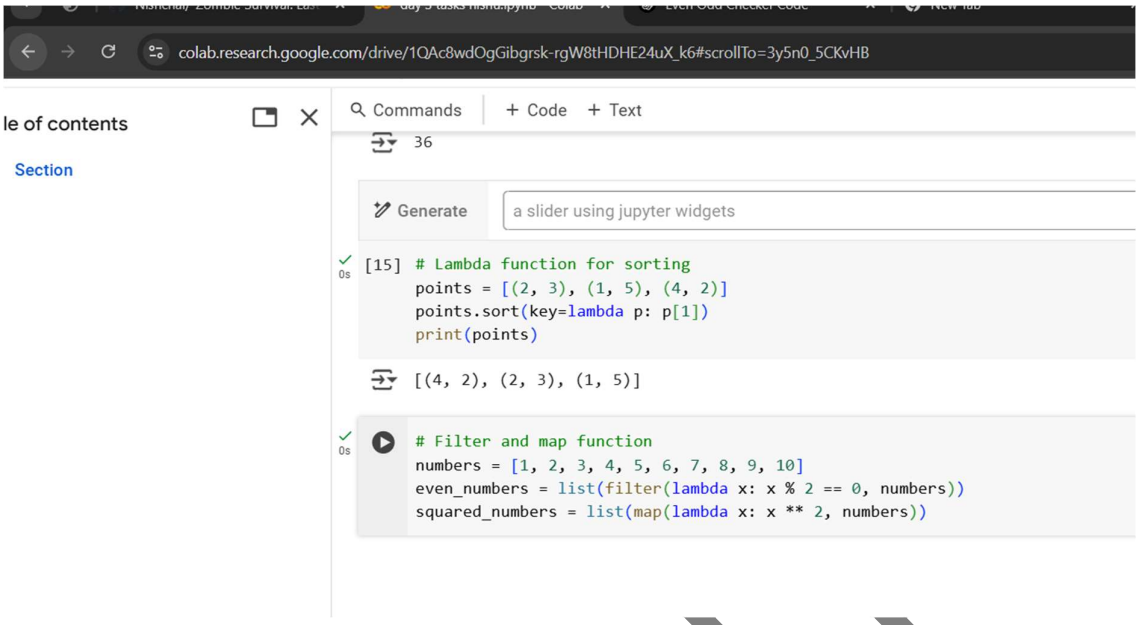
```
5.0
120
```

```
[13] # Using the random module
     import random
     print(random.randint(1, 100))
     print(random.choice(["Apple", "Banana", "Cherry"]))
```

```
37
Apple
```

```
[14] # Lambda function for squaring
     square = lambda x: x ** 2
     print(square(6))
```

```
36
```


colab.research.google.com/drive/1QAc8wdOgGibgrsk-rgW8tHDHE24uX_k6#scrollTo=3y5n0_5CKvHB
le of contents
Section
Commands
+ Code
+ Text
36
Generate
a slider using jupyter widgets
[15]
0s
# Lambda function for sorting
points = [(2, 3), (1, 5), (4, 2)]
points.sort(key=lambda p: p[1])
print(points)
[(4, 2), (2, 3), (1, 5)]
0s
# Filter and map function
numbers = [1, 2, 3, 4, 5, 6, 7, 8, 9, 10]
even_numbers = list(filter(lambda x: x % 2 == 0, numbers))
squared_numbers = list(map(lambda x: x ** 2, numbers))