String Operations in Python

1. Find the length of the string

2. Slice the string as per your choice

3. Concatenate two strings

4. Convert in to lower case in to uppercase character

5. Convert upper case into lower case characters

6. convert the character into Unicode (Ascii values)

7. convert Unicode into character

8. Check whether the given "substring" exists in the string

9. Replace the character 'k' with 'h'

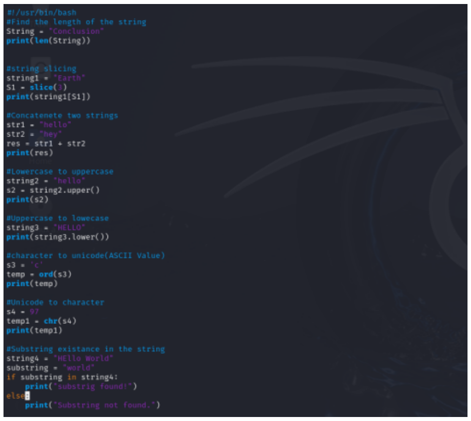
10. Pad the string with "x" at the end

11. remove leading and trailing whitespace or specified characters from the string

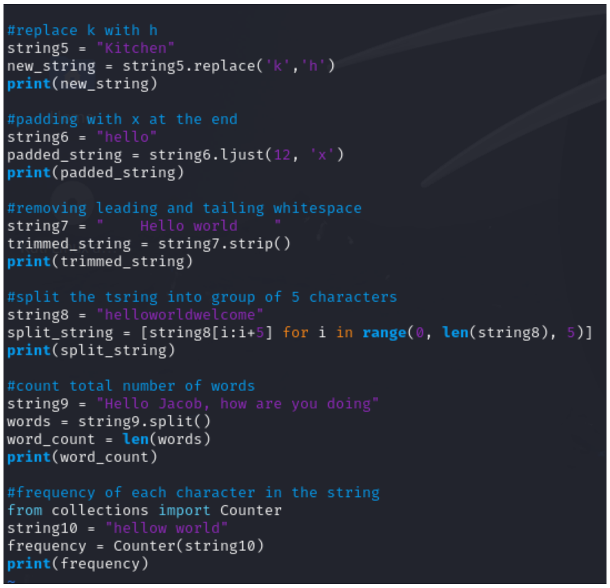
12. split the given string in to group of five characters

13. count total number of words

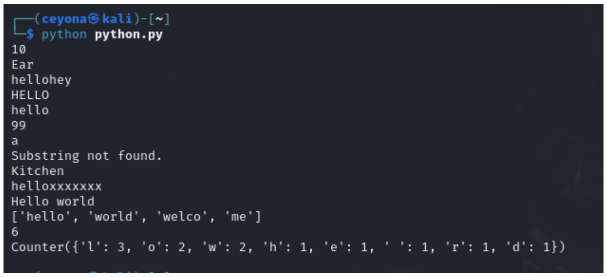
14. Find the frequency of each characters in the string



```
#!/usr/bin/bash
#Find the length of the string
String = "Conclusion"
print(len(String))


#string slicing
string1 = "Earth"
S1 = slice(3)
print(string1[S1])

#Concatenete two strings
str1 = "hello"
str2 = "hey"
res = str1 + str2
print(res)

#Lowercase to uppercase
string2 = "hello"
s2 = string2.upper()
print(s2)

#Uppercase to lowecase
string3 = "HELLO"
print(string3.lower())

#character to unicode(ASCII Value)
s3 = 'c'
temp = ord(s3)
print(temp)

#Unicode to character
s4 = 97
temp1 = chr(s4)
print(temp1)

#Substring existance in the string
string4 = "HEllo World"
substring = "world"
if substring in string4:
    print("substrig found!")
else:
    print("Substring not found.")
```



```
#replace k with h
string5 = "Kitchen"
new_string = string5.replace('k','h')
print(new_string)

#padding with x at the end
string6 = "hello"
padded_string = string6.ljust(12, 'x')
print(padded_string)

#removing leading and tailing whitespace
string7 = "    Hello world    "
trimmed_string = string7.strip()
print(trimmed_string)

#split the tsring into group of 5 characters
string8 = "helloworldwelcome"
split_string = [string8[i:i+5] for i in range(0, len(string8), 5)]
print(split_string)

#count total number of words
string9 = "Hello Jacob, how are you doing"
words = string9.split()
word_count = len(words)
print(word_count)

#frequency of each character in the string
from collections import Counter
string10 = "hellow world"
frequency = Counter(string10)
print(frequency)
~
```

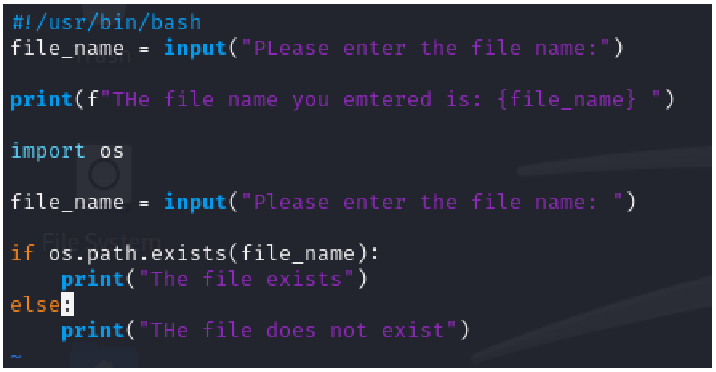


```
┌──(ceyona㉿kali)-[~]
└─$ python python.py
10
Ear
hellohey
HELLO
hello
99
a
Substring not found.
Kitchen
helloxxxxxxx
Hello world
['hello', 'world', 'welco', 'me']
6
Counter({'l': 3, 'o': 2, 'w': 2, 'h': 1, 'e': 1, ' ': 1, 'r': 1, 'd': 1})
```

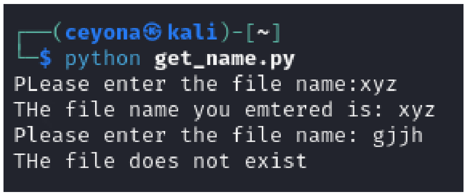
STDIN and File operators

15. get the file name from the user

16. check the file exist or not



```
#!/usr/bin/bash
file_name = input("PLease enter the file name:")

print(f"THe file name you emtered is: {file_name} ")

import os

file_name = input("Please enter the file name: ")

if os.path.exists(file_name):
    print("The file exists")
else:
    print("THe file does not exist")
~
```



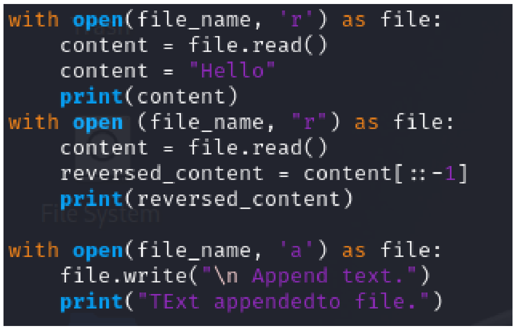
```
┌──(ceyona㉿kali)-[~]
└─$ python get_name.py
PLease enter the file name:xyz
THe file name you emtered is: xyz
Please enter the file name: gjjh
THe file does not exist
```

Looping and File handling

17. read the contents from the file

18. reverse the contents from the file

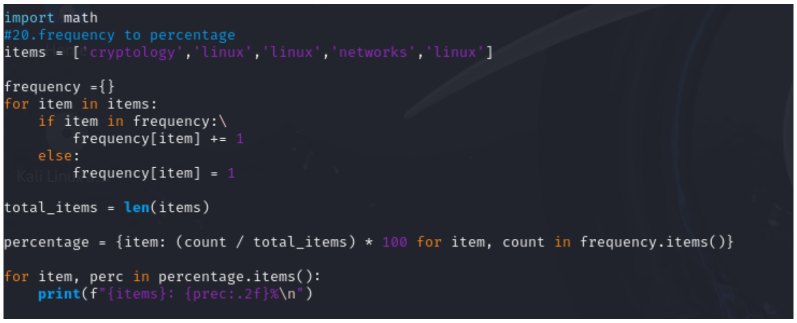
19. Write into the file



```
with open(file_name, 'r') as file:
    content = file.read()
    content = "Hello"
    print(content)
with open (file_name, "r") as file:
    content = file.read()
    reversed_content = content[::-1]
    print(reversed_content)

with open(file_name, 'a') as file:
    file.write("\n Append text.")
    print("TExt appendedto file.")
```

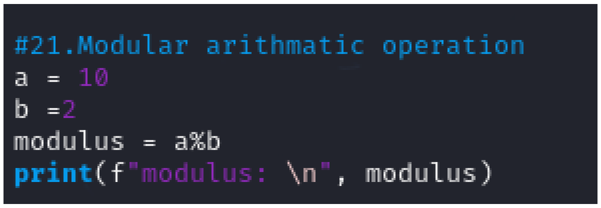
Math operations

20. convert Frequency in to percentage (continuation of 12th Question)



```
import math
#20.frequency to percentage
items = ['cryptology','linux','linux','networks','linux']

frequency ={}
for item in items:
    if item in frequency:\
        frequency[item] += 1
    else:
        frequency[item] = 1

total_items = len(items)

percentage = {item: (count / total_items) * 100 for item, count in frequency.items()}

for item, perc in percentage.items():
    print(f"{items}: {prec:.2f}%\n")
```

21. Perform modular arithmetic operation

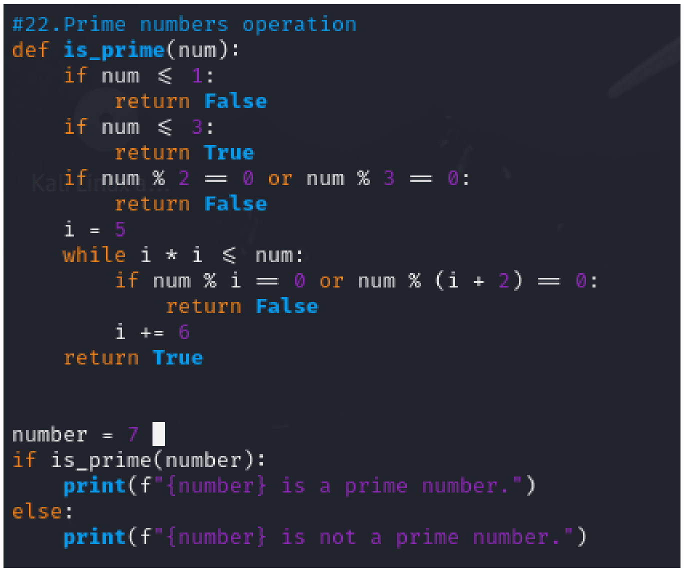


```python
#21.Modular arithmatic operation
a = 10
b =2
modulus = a%b
print(f"modulus: \n", modulus)
```

22. Find the prime numbers

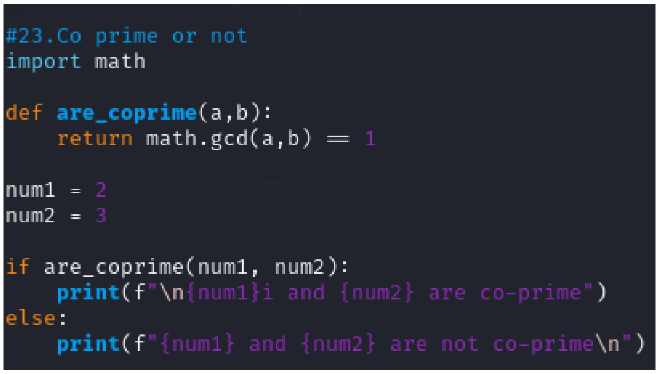
check the given number is prime or not

print the prime numbers with the given range



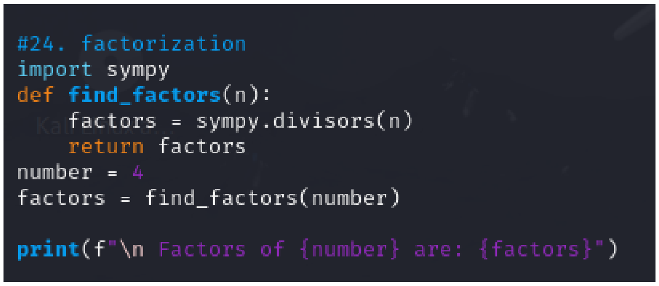
```python
#22.Prime numbers operation
def is_prime(num):
    if num <= 1:
        return False
    if num <= 3:
        return True
    if num % 2 == 0 or num % 3 == 0:
        return False
    i = 5
    while i * i <= num:
        if num % i == 0 or num % (i + 2) == 0:
            return False
        i += 6
    return True


number = 7
if is_prime(number):
    print(f"{number} is a prime number.")
else:
    print(f"{number} is not a prime number.")
```

23. Check the given two numbers are co prime or not



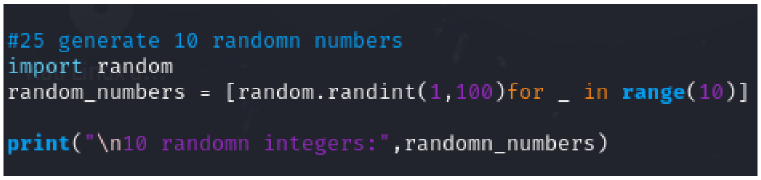
```python
#23.Co prime or not
import math

def are_coprime(a,b):
    return math.gcd(a,b) == 1

num1 = 2
num2 = 3

if are_coprime(num1, num2):
    print(f"\n{num1}i and {num2} are co-prime")
else:
    print(f"{num1} and {num2} are not co-prime\n")
```

24. find the factors for the given number (can use python library)



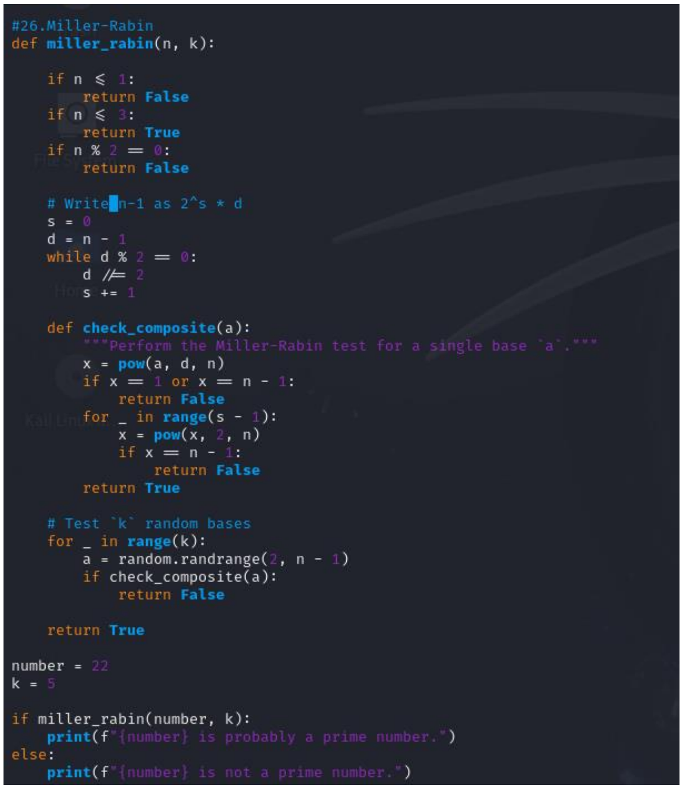
```python
#24. factorization
import sympy
def find_factors(n):
    factors = sympy.divisors(n)
    return factors
number = 4
factors = find_factors(number)

print(f"\n Factors of {number} are: {factors}")
```

25. generate 10 random numbers

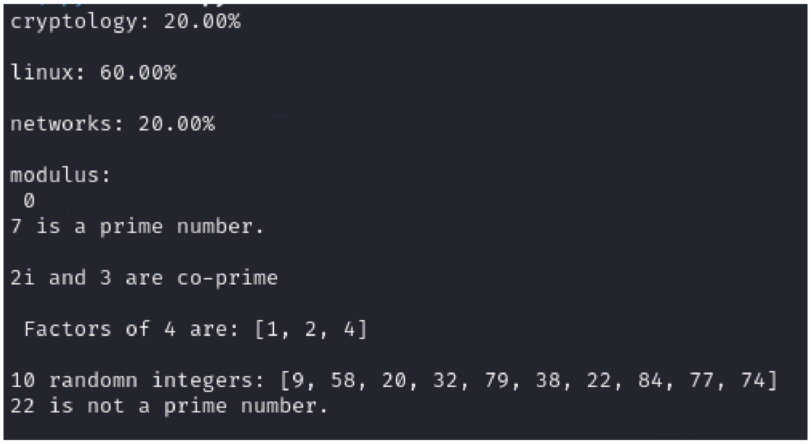


```python
#25 generate 10 randomn numbers
import random
random_numbers = [random.randint(1,100)for _ in range(10)]

print("\n10 randomn integers:",randomn_numbers)
```

26. Explore : Miller-Rabin Test (pen paper method)



```python
#26.Miller-Rabin
def miller_rabin(n, k):

    if n <= 1:
        return False
    if n <= 3:
        return True
    if n % 2 == 0:
        return False

    # Write n-1 as 2^s * d
    s = 0
    d = n - 1
    while d % 2 == 0:
        d //= 2
        s += 1

    def check_composite(a):
        """Perform the Miller-Rabin test for a single base `a`."""
        x = pow(a, d, n)
        if x == 1 or x == n - 1:
            return False
        for _ in range(s - 1):
            x = pow(x, 2, n)
            if x == n - 1:
                return False
        return True

    # Test `k` random bases
    for _ in range(k):
        a = random.randrange(2, n - 1)
        if check_composite(a):
            return False

    return True

number = 22
k = 5

if miller_rabin(number, k):
    print(f"{number} is probably a prime number.")
else:
    print(f"{number} is not a prime number.")
```



```
cryptology: 20.00%

linux: 60.00%

networks: 20.00%

modulus:
 0
7 is a prime number.

2i and 3 are co-prime

 Factors of 4 are: [1, 2, 4]

10 randomn integers: [9, 58, 20, 32, 79, 38, 22, 84, 77, 74]
22 is not a prime number.
```

"Don't worry about failures, worry about the chances you miss when you don't even try."

– JACK CANFIELD