**Python Programming Basic Assignment-17**

**Question1**.

Create a function that takes three arguments a, b, c and returns the sum of the numbers that are evenly divided by c from the range a, b inclusive.

Examples

evenly\_divisible(1, 10, 20) ➞ 0

**No number between 1 and 10 can be evenly divided by 20.**

evenly\_divisible(1, 10, 2) ➞ 30

**2 + 4 + 6 + 8 + 10 = 30**

evenly\_divisible(1, 10, 3) ➞ 18

**3 + 6 + 9 = 18**

In [34]:

**def** evenly\_divisible(a,b,c):

l**=**[]

**for** i **in** range(a,b):

**if** i**%c**==0:

l**.**append(i)

**if** len(l)**==**0:

**return** f'No number between {a} and {b} can be evenly divided by {c}.'

**return** f'Sum is {sum(l)}'

In [35]:

evenly\_divisible(1,10,20)

Out[35]:

'No number between 1 and 10 can be evenly divided by 20.'

In [36]:

evenly\_divisible(1,10,3)

Out[36]:

'Sum is 18'

In [37]:

evenly\_divisible(1,10,2)

Out[37]:

'Sum is 20'

**Question2.**

Create a function that returns True if a given inequality expression is correct and False otherwise.

Examples

correct\_signs("3 < 7 < 11") ➞ True

correct\_signs("13 > 44 > 33 > 1") ➞ False

correct\_signs("1 < 2 < 6 < 9 > 3") ➞ True

**Question3.**

Create a function that replaces all the vowels in a string with a specified character.

Examples

replace\_vowels("the aardvark", "#") ➞ "th# ##rdv#rk"

replace\_vowels("minnie mouse", "?") ➞ "m?nn?? m??s?"

replace\_vowels("shakespeare", "*") ➞ "sh*k*sp\*\*r*"

In [65]:

**class** vowel:

**def** \_\_init\_\_(self):

**pass**

**def** replace\_vowels(self,test\_str, new):

vowels **=** 'AEIOUaeiou'

**try**:

**for** i **in** vowels:

test\_str **=** test\_str**.**replace(i, new)

**return** test\_str

**except** Exception **as** e:

print(f'Error is: {e}')

In [66]:

object**=**vowel()

In [67]:

object**.**replace\_vowels('the aardvark','#')

Out[67]:

'th# ##rdv#rk'

In [68]:

object**.**replace\_vowels('minnie mouse','?')

Out[68]:

'm?nn?? m??s?'

In [69]:

object**.**replace\_vowels('shakespeare','\*')

Out[69]:

'sh\*k\*sp\*\*r\*'

**Question4.**

Write a function that calculates the factorial of a number recursively.

Examples

factorial(5) ➞ 120

factorial(3) ➞ 6

factorial(1) ➞ 1

factorial(0) ➞ 1

In [75]:

**def** factorial(num):

**if** num**==**0:

**return** 1

**else**:

**return** num**\***factorial(num**-**1)

In [79]:

factorial(5)

Out[79]:

120

In [80]:

factorial(3)

Out[80]:

6

In [81]:

factorial(1)

Out[81]:

1

In [82]:

factorial(0)

Out[82]:

1

**Question 5**

Hamming distance is the number of characters that differ between two strings.

To illustrate:

String1: "abcbba"

String2: "abcbda"

Hamming Distance: 1 - "b" vs. "d" is the only difference.

Create a function that computes the hamming distance between two strings.

Examples

hamming\_distance("abcde", "bcdef") ➞ 5

hamming\_distance("abcde", "abcde") ➞ 0

hamming\_distance("strong", "strung") ➞ 1

In [114]:

**def** hamming\_distance(a,b):

count**=**0

**try**:

**if** len(a)**==**len(b):

**try**:

**for** i **in** range(len(a)):

**if** a[i]**!=**b[i]:

count**+=**1

**except** Exception **as** e:

print(f'sub\_Error is: {e}')

**except** Exception **as** e:

print(f'main error is: {e}')

**else**:

print(count)

In [115]:

hamming\_distance('abcde','bcdef')

5

In [116]:

hamming\_distance('abcde','abcde')

0

In [117]:

hamming\_distance('strong','strung')

1