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

**Ans:**

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

sum=0

for num in range(a, b+1):

if num%c==0:

sum+=num

print(sum)

evenly\_divisible(1, 10, 3)

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

**Ans:**

def correct\_signs(expr):

result=eval(expr)

if result:

print("True")

else:

print("False")

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

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\*"

**Ans:**

def replace\_vowels(sentence, character):

vowels=['a', 'e', 'i', 'o', 'u']

for i in sentence:

if i in vowels:

sentence=str(sentence).replace(i, character)

print(sentence)

replace\_vowels("shakespeare", "\*")

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

**Ans:**

def factorial(num):

if num == 1 or num==0:

return 1

else:

return num\*factorial(num-1)

print(factorial(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

**Ans:**

def hamming\_distance(str1, str2):

count=0

for i in range(len(str1)):

if str1[i]!=str2[i]:

count+=1

print(count)

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