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: ##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.  
#inputs  
a=int(input("enter the lower limit"))  
b=int(input("enter the upper limit"))  
c=int(input("enter the divisor"))  
def sum\_evendiv(a,b,c):  
 sum=0  
 for i in range(a,b+1):  
 if i%c==0:  
 sum+=i  
 return(sum)  
  
  
def check(a,b,c):  
 if c>b:  
 print("run again and enter a value of 'c' less than 'b'")  
 else:  
 print("the sum of the numbers evenly divided by c({}) from the range a({}),b({}) inclusive".format(c,a,b),  
 sum\_evendiv(a,b,c))  
check(a,b,c)

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: # inequality function  
def check(s):  
 regex=eval(s)  
 if regex:  
 return (True)  
 else:  
 return (False)  
# printing result  
print(check("2 < 7 < 15"))  
print(check("30 > 45 > 21 > 9"))  
print(check("4 < 7 < 8< 12 > 2"))

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: ##Create a function that replaces all the vowels in a string with a specified character.  
#inputs  
str=input("enter a string : ")  
char=input("enter the special characters to replace vowels in string")  
str\_new=str  
def replace(str\_new,char):  
 c=0  
 for z in str:  
 if z=='A' or z=='a' or z=='E' or z=='e' or z=='I' or z=='i' or z=='O' or z=='o' or z=='U' or z=='u':  
 str\_new=str\_new.replace(z,char)  
 c=c+1  
 if c>1:  
 return(str\_new)  
 else:  
 return('false')  
  
ret=replace(str\_new,char)  
print("the original string : " , str)  
if ret=='false':  
 print("No vowels present in the given string ")  
else:  
 print("the new string : ",ret)

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 recur\_factorial(n):  
 if n == 1:  
 return n  
 else:  
 return n\*recur\_factorial(n-1)  
# take input from the user  
num = int(input("Enter a number: "))  
# check is the number is negative  
if num < 0:  
 print("Sorry, factorial does not exist for negative numbers")  
elif num == 0:  
 print("The factorial of 0 is 1")  
else:  
 print("The factorial of",num,"is",recur\_factorial(num))

**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: ##a function that computes the hamming distance between two strings  
#inputs  
str1=input("enter string 1: ")  
str2=input("enter string 2: ")  
  
def hamming(str1,str2):  
 d=0  
 for i in range(len(str1)):  
 if str1[i]!=str2[i]:  
 d+=1  
 return(d)  
def check(str1,str2):  
 if len(str1)==len(str2):  
 print("the hamming distance for the given strings : ",hamming(str1,str2))  
 else:  
 print("rerun and enter strings of same length")  
check(str1,str2)