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.

import logging as lg

# importing logging so every function call of

lg.basicConfig(filename ='C:\\Users\\Home\\Johns python talent\\logging\\testlog1.log', level =lg.INFO , format = '%(asctime)s %(message)s')

def evenly\_divisible(start,end,divider):

if divider >= end:

print("no number between", start, "and", end, "can be evenly divided by", divider)

else:

sum\_all = 0

for i in range(start,end+1):

if i%divider == 0:

sum\_all += i

print(sum\_all)

return sum\_all

try:

evenly\_divisible(1, 10, 20)

evenly\_divisible(1, 10, 2)

evenly\_divisible(1, 10, 3)

lg.info("""Class evenly\_divisible() has been called has been called""")

except Exception as e:

print("There was an error called: ",e)

else:

pass

finally:

pass

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.

import logging as lg

# importing logging so every function call of

lg.basicConfig(filename ='C:\\Users\\Home\\Johns python talent\\logging\\testlog1.log', level =lg.INFO , format = '%(asctime)s %(message)s')

def correct\_signs(string\_condition):

print(eval(string\_condition))

try:

correct\_signs("3 < 7 < 11")

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

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

lg.info("""Class correct\_signs() has been called has been called""")

except Exception as e:

print("There was an error called: ",e)

else:

pass

finally:

pass

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.

import logging as lg

# importing logging so every function call of

lg.basicConfig(filename ='C:\\Users\\Home\\Johns python talent\\logging\\testlog1.log', level =lg.INFO , format = '%(asctime)s %(message)s')

def replace\_vowels(main\_string, replacer):

final\_string = ""

for i in main\_string:

if i == 'a'or i =='e'or i =='i' or i =='o'or i =='u' or i =='A'or i =='E'or i =='I'or i =='O' or i =='U':

final\_string += replacer

else:

final\_string += i

print(final\_string)

try:

replace\_vowels("the aardvark", "#")

replace\_vowels("minnie mouse", "?")

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

lg.info("""Class replace\_vowels() has been called has been called""")

except Exception as e:

print("There was an error called: ",e)

else:

pass

finally:

pass

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.

import logging as lg

# importing logging so every function call of

lg.basicConfig(filename ='C:\\Users\\Home\\Johns python talent\\logging\\testlog1.log', level =lg.INFO , format = '%(asctime)s %(message)s')

def factorial(n):

if n == 0:

return 1

if n == 1:

return 1

else:

fact = n\*factorial(n-1)

return fact

try:

j = factorial(5)

print(j)

lg.info("""Class factorial() has been called has been called""")

except Exception as e:

print("There was an error called: ",e)

else:

pass

finally:

pass

**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.

import logging as lg

# importing logging so every function call of

lg.basicConfig(filename ='C:\\Users\\Home\\Johns python talent\\logging\\testlog1.log', level =lg.INFO , format = '%(asctime)s %(message)s')

def hamming\_distance(first\_string,second\_string):

count = 0

for i,j in zip(first\_string,second\_string):

if i != j:

count+=1

print(count)

try:

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

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

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

lg.info("""Class factorial() has been called has been called""")

except Exception as e:

print("There was an error called: ",e)

else:

pass

finally:

pass