Question1

Create a function that takes a string and returns a string in which each character is repeated once.

**Examples**

double\_char("String") ➞ "SSttrriinngg"

double\_char("Hello World!") ➞ "HHeelllloo WWoorrlldd!!"

double\_char("1234!\_ ") ➞ "11223344!!\_\_ "

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 double\_char(my\_string):

for i in my\_string:

print(i\*2, end="")

try:

double\_char("Hello World!")

lg.info("""Class double\_char() 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 reverses a boolean value and returns the string "boolean expected" if another variable type is given.

### Examples

reverse(True) ➞ False

reverse(False) ➞ True

reverse(0) ➞ "boolean expected"

reverse(None) ➞ "boolean expected"

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 reverse(my\_bool):

if type(my\_bool) != bool:

print("boolean expected")

if type(my\_bool) == bool:

print(not my\_bool)

try:

reverse(True)

reverse(False)

reverse(0)

reverse(None)

lg.info("""Class reverse() 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 returns the **thickness (in meters)** of a piece of paper after folding it n number of times. The paper starts off with a thickness of **0.5mm**.

### Examples

num\_layers(1) ➞ "0.001m"

# Paper folded once is 1mm (equal to 0.001m)

num\_layers(4) ➞ "0.008m"

# Paper folded 4 times is 8mm (equal to 0.008m)

num\_layers(21) ➞ "1048.576m"

# Paper folded 21 times is 1048576mm (equal to 1048.576m)

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 num\_layers(no\_of\_folds):

print(((2\*\*no\_of\_folds)\*0.5)/1000,"m")

return ((2\*\*no\_of\_folds)\*0.5)/1000

try:

num\_layers(1)

num\_layers(4)

num\_layers(21)

lg.info("""Class num\_layers() has been called has been called""")

except Exception as e:

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

else:

pass

finally:

pass

Question4

Create a function that takes a single string as argument and returns an ordered list containing the indices of all capital letters in the string.

### Examples

index\_of\_caps("eDaBiT") ➞ [1, 3, 5]

index\_of\_caps("eQuINoX") ➞ [1, 3, 4, 6]

index\_of\_caps("determine") ➞ []

index\_of\_caps("STRIKE") ➞ [0, 1, 2, 3, 4, 5]

index\_of\_caps("sUn") ➞ [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 index\_of\_caps(my\_string):

index = 0

my\_indexes = []

for i in my\_string:

if i.isupper() == True:

my\_indexes.append(index)

index += 1

print(my\_indexes)

try:

index\_of\_caps("eDaBiT")

index\_of\_caps("eQuINoX")

index\_of\_caps("determine")

index\_of\_caps("STRIKE")

index\_of\_caps("sUn")

lg.info("""Class index\_of\_caps() has been called has been called""")

except Exception as e:

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

else:

pass

finally:

pass

Question5

Using list comprehensions, create a function that finds all even numbers from 1 to the given number.

### Examples

find\_even\_nums(8) ➞ [2, 4, 6, 8]

find\_even\_nums(4) ➞ [2, 4]

find\_even\_nums(2) ➞ [2]

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 find\_even\_nums(number):

even =[i for i in range(2,number+1) if i%2 ==0]

print(even)

return even

try:

find\_even\_nums(8)

find\_even\_nums(4)

find\_even\_nums(2)

lg.info("""Class find\_even\_numbers() has been called has been called""")

except Exception as e:

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

else:

pass

finally:

pass