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!!\_\_ "

def double\_char(n):

res= ''

for i in n:

res=res + i \* 2

print(res)

double\_char("String")

double\_char("Hello World!")

double\_char("1234!\_ ")

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"

def reverse(n):

if n is True:

print(False)

elif n is False:

print(True)

elif n == 0 or n is None:

print("boolean expected")

else:

print('ERROR..!')

# Test cases

reverse(True) # ➞ False

reverse(False) # ➞ True

reverse(0) # ➞ "boolean expected"

reverse(None) # ➞ "boolean expected"

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)

def num\_layers(n):

initial\_thickness\_mm = 0.5 # Paper's initial thickness in millimeters

thickness\_mm = initial\_thickness\_mm \* (2 \*\* n) # Thickness after n folds

thickness\_m = thickness\_mm / 1000 # Convert from mm to meters

return f"{thickness\_m:.3f}m" # Return the result formatted to 3 decimal places

# Test cases

print(num\_layers(1)) # ➞ "0.001m"

print(num\_layers(4)) # ➞ "0.008m"

print(num\_layers(21)) # ➞ "1048.576m"

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]

def index\_of\_caps(n):

l=[]

for i in range(len(n)):

if n[i].isupper():

l.append(i)

print(l)

index\_of\_caps("eDaBiT")

index\_of\_caps("eQuINoX")

index\_of\_caps("determine")

index\_of\_caps("STRIKE")

index\_of\_caps("sUn")

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]

def find\_even\_nums(n):

l=[]

for i in range(1,n+1):

if i %2 == 0:

l.append(i)

print(l)

find\_even\_nums(8)#[2, 4, 6, 8]

find\_even\_nums(4)# [2, 4]

find\_even\_nums(2) # [2]