**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(string):

doubled\_string = ""

for char in string:

doubled\_string += char \* 2

return doubled\_string

# Example Usage

print(double\_char("String")) # Output: "SSttrriinngg"

print(double\_char("Hello World!")) # Output: "HHeelllloo WWoorrlldd!!"

print(double\_char("1234!\_ ")) # Output: "11223344!!\_\_ "

**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(boolean):

if isinstance(boolean, bool):

return not boolean

else:

return "boolean expected"

# Example Usage

print(reverse(True)) # Output: False

print(reverse(False)) # Output: True

print(reverse(0)) # Output: "boolean expected"

print(reverse(None)) # Output: "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):

thickness\_mm = 0.5 \* (2 \*\* n)

thickness\_m = thickness\_mm / 1000

return f"{thickness\_m:.3f}m"

# Example Usage

print(num\_layers(1)) # Output: "0.001m"

print(num\_layers(4)) # Output: "0.008m"

print(num\_layers(21)) # Output: "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(word):

indices = []

for i in range(len(word)):

if word[i].isupper():

indices.append(i)

return indices

# Example Usage

print(index\_of\_caps("eDaBiT")) # Output: [1, 3, 5]

print(index\_of\_caps("eQuINoX")) # Output: [1, 3, 4, 6]

print(index\_of\_caps("determine")) # Output: []

print(index\_of\_caps("STRIKE")) # Output: [0, 1, 2, 3, 4, 5]

print(index\_of\_caps("sUn")) # Output: [1]

**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):

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

# Example Usage

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

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

print(find\_even\_nums(2)) # Output: [2]a