1. def double\_char(in\_string):

out\_string = ''

for ele in in\_string:

out\_string += ele\*2

return out\_string

print(f'➞ {double\_char("String")}')

print(f'➞ {double\_char("Hello World!")}')

print(f'➞ {double\_char("1234!\_")}')

Output:

➞ SSttrriinngg

➞ HHeelllloo WWoorrlldd!!

➞ 11223344!!\_\_

1. def reverse(in\_bool):

if type(in\_bool) == bool:

return not in\_bool

else:

return "Boolean Expected"

print(f'reverse(True) ➞ {reverse(True)}')

print(f'reverse(False) ➞ {reverse(False)}')

print(f'reverse(0) ➞ {reverse(0)}')

print(f'reverse(None) ➞ {reverse(None)}')

Output:

reverse(True) ➞ False

reverse(False) ➞ True

reverse(0) ➞ Boolean Expected

reverse(None) ➞ Boolean Expected

1. def num\_layers(in\_num):

out\_num = 0.5

for ele in range(in\_num):

out\_num \*= 2

print(f'Output ➞ {out\_num/1000}m')

num\_layers(1)

num\_layers(4)

num\_layers(21)

Output:

Output ➞ 0.001m

Output ➞ 0.008m

Output ➞ 1048.576m

1. def index\_of\_caps(in\_string):

out\_string = []

for ele in in\_string:

if ele.isupper():

out\_string.append(in\_string.index(ele))

print(f'{in\_string} ➞ {out\_string}')

index\_of\_caps("eDaBiT")

index\_of\_caps("eQuINoX")

index\_of\_caps("determine")

index\_of\_caps("STRIKE")

index\_of\_caps("sUn")

Output:

eDaBiT ➞ [1, 3, 5]

eQuINoX ➞ [1, 3, 4, 6]

determine ➞ []

STRIKE ➞ [0, 1, 2, 3, 4, 5]

sUn ➞ [1]

1. def find\_even\_nums(in\_num):

out\_list = [i for i in range(1,in\_num+1) if i%2 == 0]

print(f'Output ➞ {out\_list}')

find\_even\_nums(8)

find\_even\_nums(4)

find\_even\_nums(2)

Output:

Output ➞ [2, 4, 6, 8]

Output ➞ [2, 4]

Output ➞ [2]