1. def secret(in\_string):

in\_string\_clone = in\_string

in\_string = in\_string.split(".")

output = f'<{in\_string[0]} class="{" ".join(in\_string[1:])}"></{in\_string[0]}>'

print(f'secret("{in\_string\_clone}") ➞ {output}')

secret("p.one.two.three")

secret("p.one")

secret("p.four.five")

Output:

secret("p.one.two.three") ➞ <p class="one two three"></p>

secret("p.one") ➞ <p class="one"></p>

secret("p.four.five") ➞ <p class="four five"></p>

1. import re

def count\_lone\_ones(in\_num):

pattern = r"(?<!1)1(?!1)"

output = re.findall(pattern,str(in\_num))

print(f'coint\_lone\_ones({in\_num}) ➞ {len(output)}')

count\_lone\_ones(101)

count\_lone\_ones(1191)

count\_lone\_ones(1111)

count\_lone\_ones(462)

Output:

coint\_lone\_ones(101) ➞ 2

coint\_lone\_ones(1191) ➞ 1

coint\_lone\_ones(1111) ➞ 0

coint\_lone\_ones(462) ➞ 0

1. def printGrid(in\_one,in\_two):

output = []

for ele\_1 in range(in\_one):

temp = []

for ele\_2 in range(in\_two):

temp.append(ele\_1+(in\_one\*ele\_2)+1)

output.append(temp)

print(f'printGrid{in\_one,in\_two} ➞ {output}')

printGrid(3, 6)

printGrid(5, 3)

printGrid(4, 1)

Output:

printGrid(3, 6) ➞ [[1, 4, 7, 10, 13, 16], [2, 5, 8, 11, 14, 17], [3, 6, 9, 12, 15, 18]]

printGrid(5, 3) ➞ [[1, 6, 11], [2, 7, 12], [3, 8, 13], [4, 9, 14], [5, 10, 15]]

printGrid(4, 1) ➞ [[1], [2], [3], [4]]

1. def min\_miss\_pos(in\_list):

in\_list\_clone = in\_list.copy()

in\_list = sorted(in\_list)

output = -1

for ele in range(1,max(in\_list)+1):

if ele not in in\_list:

output = ele

break

print(f'min\_miss\_pos({in\_list\_clone}) ➞ {in\_list} ➞ {output}')

min\_miss\_pos([-2, 6, 4, 5, 7, -1, 1, 3, 6, -2, 9, 10, 2, 2])

min\_miss\_pos([5, 9, -2, 0, 1, 3, 9, 3, 8, 9])

min\_miss\_pos([0, 4, 4, -1, 9, 4, 5, 2, 10, 7, 6, 3, 10, 9])

Output:

min\_miss\_pos([-2, 6, 4, 5, 7, -1, 1, 3, 6, -2, 9, 10, 2, 2]) ➞ [-2, -2, -1, 1, 2, 2, 3, 4, 5, 6, 6, 7, 9, 10] ➞ 8

min\_miss\_pos([5, 9, -2, 0, 1, 3, 9, 3, 8, 9]) ➞ [-2, 0, 1, 3, 3, 5, 8, 9, 9, 9] ➞ 2

min\_miss\_pos([0, 4, 4, -1, 9, 4, 5, 2, 10, 7, 6, 3, 10, 9]) ➞ [-1, 0, 2, 3, 4, 4, 4, 5, 6, 7, 9, 9, 10, 10] ➞ 1

1. def pizza\_points(in\_dict,min\_order,min\_price):

output = []

for customer in in\_dict.keys():

if len([order\_price for order\_price in in\_dict[customer] if order\_price > min\_price]) > min\_order:

output.append(customer)

print(f'pizza\_points{"customers",min\_order,min\_price} ➞ {output}')

customers = {

"Batman": [22, 30, 11, 17, 15, 52, 27, 12],

"Spider-Man": [5, 17, 30, 33, 40, 22, 26, 10, 11, 45]

}

pizza\_points(customers, 5, 20)

pizza\_points(customers, 3, 10)

pizza\_points(customers, 5, 100)

Output:

pizza\_points('customers', 5, 20) ➞ ['Spider-Man']

pizza\_points('customers', 3, 10) ➞ ['Batman', 'Spider-Man']

pizza\_points('customers', 5, 100) ➞ []