Question1

Create a function that takes three integer arguments (a, b, c) and returns the amount of integers which are of equal value.

**Examples**

equal(3, 4, 3) ➞ 2

equal(1, 1, 1) ➞ 3

equal(3, 4, 1) ➞ 0

**Notes**

Your function must return 0, 2 or 3.

Ans 1

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 equal(a1,a2,a3):

l = [a1,a2,a3]

es = []

k = 0

for j in range(len(l)):

for i in l:

if l[j] == i:

k += 1

es.append(k)

k = 0

if max(es) == 1:

print( "--> 0")

if max(es) == 2:

print("--> 2")

if max(es) == 3:

print("--> 3")

try:

equal(3, 4, 3)

equal(1, 1, 1)

equal(3, 4, 1)

lg.info("""function equal() has been called""")

except Exception as e:

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

else:

pass

finally:

pass

Question2

Write a function that converts a **dictionary** into a **list** of keys-values **tuples**.

### Examples

dict\_to\_list({

"D": 1,

"B": 2,

"C": 3

}) ➞ [("B", 2), ("C", 3), ("D", 1)]

dict\_to\_list({

"likes": 2,

"dislikes": 3,

"followers": 10

}) ➞ [("dislikes", 3), ("followers", 10), ("likes", 2)]

### Notes

Return the elements in the list in alphabetical order.

Ans.2

Code =

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 dict\_to\_list(my\_dict):

print([i for i in my\_dict.items()])

try:

dict\_to\_list({

"D": 1,

"B": 2,

"C": 3

})

dict\_to\_list({

"likes": 2,

"dislikes": 3,

"followers": 10})

lg.info("""Function dict\_to\_list() has been called""")

except Exception as e:

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

else:

pass

finally:

pass

Question3

Write a function that creates a dictionary with each **(key, value)** pair being the **(lower case, upper case)** versions of a letter, respectively.

### Examples

mapping(["p", "s"]) ➞ { "p": "P", "s": "S" }

mapping(["a", "b", "c"]) ➞ { "a": "A", "b": "B", "c": "C" }

mapping(["a", "v", "y", "z"]) ➞ { "a": "A", "v": "V", "y": "Y", "z": "Z" }

### Notes

All of the letters in the input list will always be lowercase.

Ans 3

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 mapping(\*args):

main\_list = []

main\_list += args

print({i.lower():i.upper() for i in main\_list})

print("Kindly enter string characters separated by commas ',' in mapping() function")

# defining a function to print our required string

try:

mapping('a','t','u')

lg.info('Function mapping() has been called')

except exception as e:

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

else:

pass

finally:

pass

Question4

Write a function, that replaces all vowels in a string with a specified vowel.

### Examples

vow\_replace("apples and bananas", "u") ➞ "upplus und bununus"

vow\_replace("cheese casserole", "o") ➞ "chooso cossorolo"

vow\_replace("stuffed jalapeno poppers", "e") ➞ "steffed jelepene peppers"

### Notes

All words will be lowercase. Y is not considered a vowel.

Ans 4

Code =

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 vow\_replace(sentence, user\_choice):

empty\_sentence = ""

vowels = ['a', 'A','e', 'E','i', 'I', 'o', 'O' ]

l = list(sentence)

j = -1

for i in l:

j += 1

if i in vowels:

l[j] = user\_choice

for k in l:

empty\_sentence += k

print(empty\_sentence)

try:

vow\_replace("apples and bananas", "u")

vow\_replace("cheese casserole", "o")

vow\_replace("stuffed jalapeno poppers", "e")

lg.info("""Function dict\_to\_list() has been called""")

except Exception as e:

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

else:

pass

finally:

pass

Question5

Create a function that takes a string as input and capitalizes a letter if its ASCII code is even and returns its lower case version if its ASCII code is odd.

### Examples

ascii\_capitalize("to be or not to be!") ➞ "To Be oR NoT To Be!"

ascii\_capitalize("THE LITTLE MERMAID") ➞ "THe LiTTLe meRmaiD"

ascii\_capitalize("Oh what a beautiful morning.") ➞ "oH wHaT a BeauTiFuL moRNiNg."

Ans 5

Code =

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 ascii\_capitalize(my\_sentence):

h = list(my\_sentence.lower())

empty = ""

j = -1

for i in h:

j += 1

if ord(i)%2 == 0:

h[j] = i.upper()

for k in h:

empty += k

print(empty)

try:

ascii\_capitalize("to be or not to be!")

ascii\_capitalize("THE LITTLE MERMAID")

ascii\_capitalize("Oh what a beautiful morning.")

lg.info("""Function dict\_to\_list() has been called""")

except Exception as e:

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

else:

pass

finally:

pass