**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.**

We can solve this problem using a simple approach where we check all three integers to see if they are equal or not.

Algorithm:

1. If all the three integers are equal, return 3.

2. If two of the integers are equal, return 2.

3. Otherwise, return 0.

Let's write the code to implement this algorithm in Python:

def equal(a, b, c):

if a == b == c:

return 3

elif a == b or b == c or c == a:

return 2

else:

return 0

#test the function

print(equal(3, 4, 3)) #2

print(equal(1, 1, 1)) #3

print(equal(3, 4, 1)) #0

The output shows that the function is working as expected.

**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.**

def dict\_to\_list(d):

return sorted(d.items())

print(dict\_to\_list({"D": 1, "B": 2, "C": 3}))

# Output: [('B', 2), ('C', 3), ('D', 1)]

print(dict\_to\_list({"likes": 2, "dislikes": 3, "followers": 10}))

# Output: [('dislikes', 3), ('followers', 10), ('likes', 2)]

**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.**

def mapping(letters):

return {l: l.upper() for l in letters}

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.

def vow\_replace(txt, vowel):

vowels = "aeiou"

return "".join(vowel if c in vowels else c for c in txt)

>>> 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'

**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."**

def ascii\_capitalize(txt):

result = ''

for c in txt:

if ord(c) % 2 == 0:

result += c.upper()

else:

result += c.lower()

return result

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

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

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