

python_basic_programming_25

May 20, 2023

[]: 1. 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.

```
[1]: def equal(a,b,c):
      if a==b==c:
          print(f'{a,b,c} {3}')
      elif a==b or b==c:
          print(f'{a,b,c} {2}')
      else:
          print(f'{a,b,c} {0}')
```

```
equal(3, 4, 3)
equal(1, 1, 1)
equal(3, 4, 1)
```

```
(3, 4, 3) 0
(1, 1, 1) 3
(3, 4, 1) 0
```

[]: 2. 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.

```
[2]: def dict_to_list(in_dict):
      out_list = []
      for keys, values in in_dict.items():
```

```

        out_list.append((keys,values))
    print(f'{in_dict}  {out_list}')

dict_to_list({"D": 1,"B": 2,"C": 3})
dict_to_list({"likes": 2,"dislikes": 3,"followers": 10})

```

```

{'D': 1, 'B': 2, 'C': 3}  [('D', 1), ('B', 2), ('C', 3)]
{'likes': 2, 'dislikes': 3, 'followers': 10}  [('likes', 2), ('dislikes', 3),
('followers', 10)]

```

[]: 3. 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.

```

[3]: def mapping(in_list):
    out_dict = {}
    for ele in in_list:
        out_dict[ele] = ele.upper()
    print(f'{in_list}  {out_dict}')

```

```

mapping(["p", "s"])
mapping(["a", "b", "c"])
mapping(["a", "v", "y", "z"])

```

```

['p', 's']  {'p': 'P', 's': 'S'}
['a', 'b', 'c']  {'a': 'A', 'b': 'B', 'c': 'C'}
['a', 'v', 'y', 'z']  {'a': 'A', 'v': 'V', 'y': 'Y', 'z': 'Z'}

```

[]: 4. 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.

```

[4]: def vow_replace(in_string,vow_char):
    vowels = ['a','e','i','o','u']
    out_string = ''
    for ele in in_string:

```

```

        if ele in vowels:
            out_string += vow_char
        else:
            out_string += ele
    print(f'{in_string} {out_string}')

```

```

vow_replace("apples and bananas", "u")
vow_replace("cheese casserole", "o")
vow_replace("stuffed jalapeno poppers", "e")

```

```

apples and bananas  upplus und bununus
cheese casserole   chooso cossorolo
stuffed jalapeno poppers  steffed jelepene peppers

```

```

[ ]: 5.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.
      ↳

```

```

[5]: def ascii_capitalize(in_string):
        out_string = ''
        for ele in in_string.lower():
            if (ord(ele)%2 == 0):
                out_string += ele.upper()
            else:
                out_string += ele
        print(f'{in_string} {out_string}')

ascii_capitalize("to be or not to be!")
ascii_capitalize("THE LITTLE MERMAID")
ascii_capitalize("Oh what a beautiful morning.")

```

```

to be or not to be!  To Be oR NoT To Be!
THE LITTLE MERMAID  The LiTTLe meRmaiD
Oh what a beautiful morning.  oH wHaT a BeauTiFuL moRniNg.

```

```

[ ]:

```