

# Most Common Name?

# Most Common Name? 1/2

marquard

cwen

cwen

zhen

marquard

zhen

csev

zhen

csev

zhen

csev

marquard

zhen

# Most Common Name? 2/2

marquard

cwen

cwen

zhen

zhen

csev

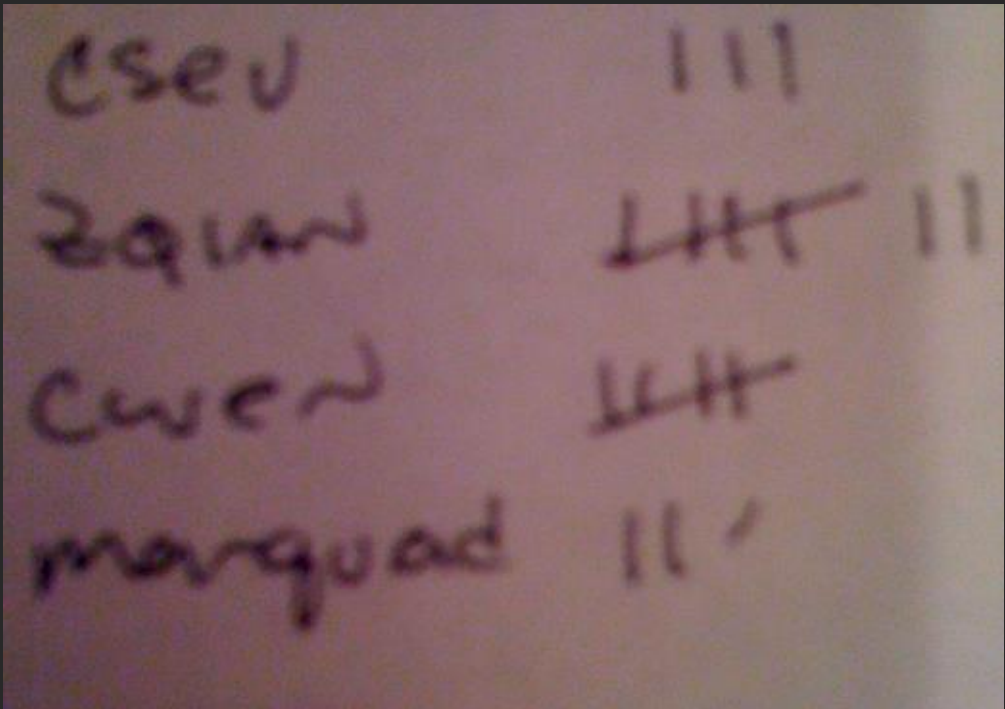
csev

marquard

zhen

csev

zhen



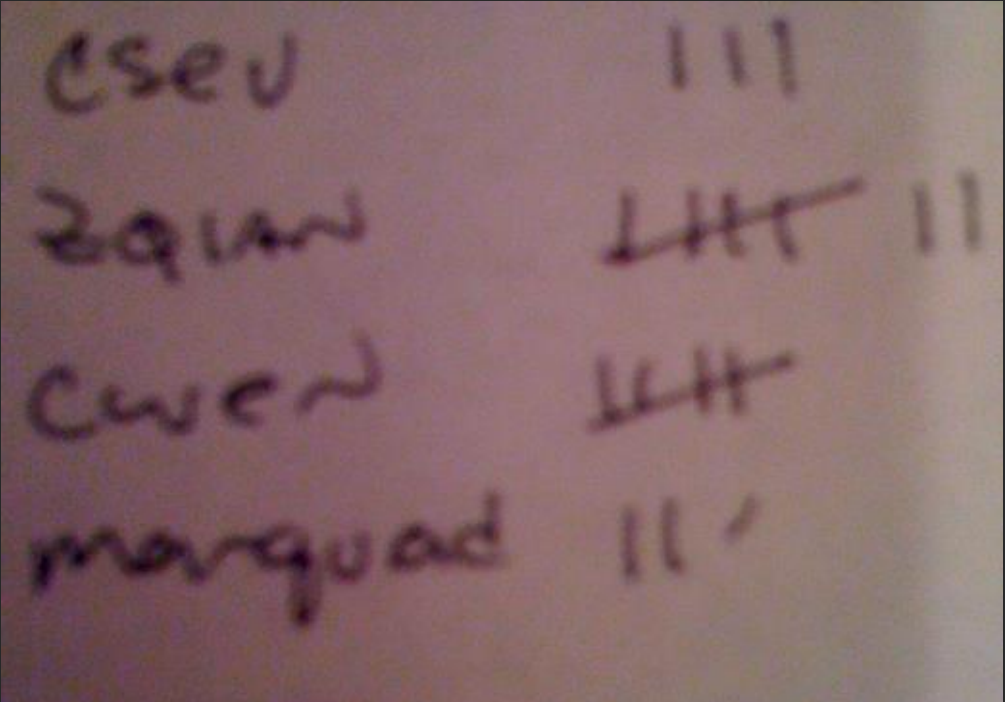
# Many Counters with a Dictionary

One common use of dictionaries is **counting** how often we “see” something

```
>>> ccc = dict()
>>> ccc['csev'] = 1
>>> ccc['cwen'] = 1
>>> print(ccc)
{'csev': 1, 'cwen': 1}
>>> ccc['cwen'] = ccc['cwen'] + 1
>>> print(ccc)
{'csev': 1, 'cwen': 2}
```

Key

Value



csev	
zqian	<del>    </del>
cwen	
marquard	

# Dictionary Tracebacks

- It is an **error** to reference a key which is not in the dictionary
- We can use the **in** operator to see if a key is in the dictionary

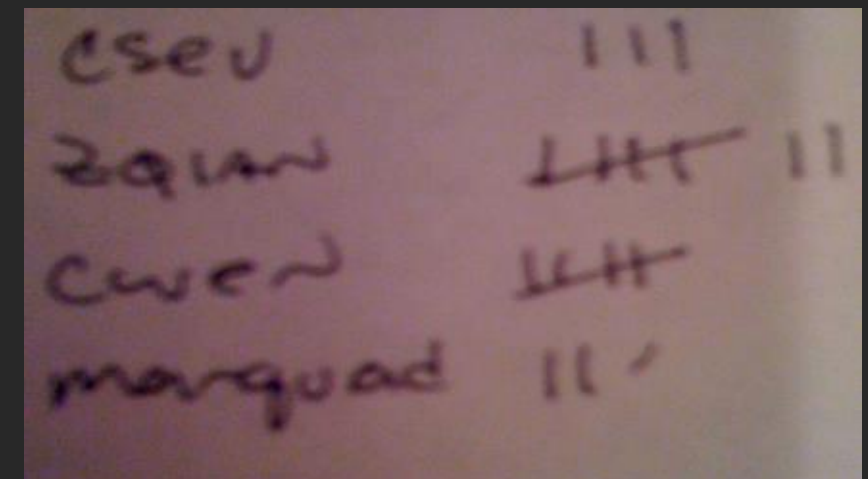
```
>>> ccc = dict()
>>> print(ccc['csev'])
Traceback (most recent call last):
  File "<stdin>", line 1, in <module>
KeyError: 'csev'
>>> 'csev' in ccc
False
```

# When We See a New Name

When we encounter a new name, we need to add a new entry in the **dictionary** and if this the second or later time we have seen the **name**, we simply add one to the count in the **dictionary** under that **name**

```
counts = dict()
names = ['csev', 'cwen', 'csev', 'zqian', 'cwen']
for name in names:
    if name not in counts:
        counts[name] = 1
    else:
        counts[name] = counts[name] + 1
print(counts)
```

**{'csev': 2, 'cwen': 2, 'zqian': 1}**



# The `get` Method for Dictionaries

The pattern of checking to see if a `key` is already in a dictionary and assuming a default value if the `key` is not there is so common that there is a `method` called `get()` that does this for us

Default value if key does not exist  
(and no Traceback).

```
if name in counts:  
    x = counts[name]  
else :  
    x = 0
```

```
x = counts.get(name, 0)
```

```
{'csev': 2, 'cwen': 2 , 'zqian': 1}
```

# Simplified Counting with `get()` 1/2

We can use `get()` and provide a **default value of zero** when the **key** is not yet in the dictionary - and then just add one

```
counts = dict()
names = ['csev', 'cwen', 'csev', 'zqian', 'cwen']
for name in names:
    counts[name] = counts.get(name, 0) + 1
print(counts)
```

Default



`{'csev': 2, 'cwen': 2, 'zqian': 1}`



# Simplified Counting with `get()` 2/2

```
counts = dict()
names = ['csev', 'cwen', 'csev', 'zqian', 'cwen']
for name in names:
    counts[name] = counts.get(name, 0) + 1
print(counts)
```



Counting is Marvelous  
– the Count on Sesame Street