

Fundamentos de Programación

Grado en Ingeniería del Software

21/09/22

Antecedentes

- Parametro por defecto
- Paso de función como parámetro
- Funciones lambda
- Construcción de diccionarios
- Ordenaciones alternativas

Índice

- Counter
- defaultdict
- Agrupación y transformación

COUNTER

Python

```
>>> from collections import Counter

>>> # Use a string as an argument
>>> Counter("mississippi")
Counter({'i': 4, 's': 4, 'p': 2, 'm': 1})

>>> # Use a list as an argument
>>> Counter(list("mississippi"))
Counter({'i': 4, 's': 4, 'p': 2, 'm': 1})
```

Python

>>>

```
>>> from collections import Counter
>>> sales = Counter(banana=15, tomato=4, apple=39, orange=30)

>>> # The most common object
>>> sales.most_common(1)
[('apple', 39)]

>>> # The two most common objects
>>> sales.most_common(2)
[('apple', 39), ('orange', 30)]

>>> # All objects sorted by count
>>> sales.most_common()
[('apple', 39), ('orange', 30), ('banana', 15), ('tomato', 4)]

>>> sales.most_common(None)
[('apple', 39), ('orange', 30), ('banana', 15), ('tomato', 4)]

>>> sales.most_common(20)
[('apple', 39), ('orange', 30), ('banana', 15), ('tomato', 4)]
```

DEFAULTDICT

Python3



```
# Python program to demonstrate  
# default_factory argument of  
# defaultdict
```



```
from collections import defaultdict
```



```
# Defining the dict and passing  
# lambda as default_factory argument  
d = defaultdict(lambda: "Not Present")  
d["a"] = 1  
d["b"] = 2  
  
print(d["a"])  
print(d["b"])  
print(d["c"])
```



```
1  
2  
Not Present
```


Python3



```
# Python program to demonstrate  
# defaultdict
```



```
from collections import defaultdict
```



```
# Defining the dict  
d = defaultdict(int)  
  
L = [1, 2, 3, 4, 2, 4, 1, 2]  
  
# Iterate through the list  
# for keeping the count  
for i in L:  
    # The default value is 0  
    # so there is no need to  
    # enter the key first  
    d[i] += 1  
  
print(d)
```

Output:

```
defaultdict(<class 'int'>, {1: 2, 2: 3, 3: 1, 4: 2})
```

AGRUPACIÓN Y TRANSFORMACIÓN

¿Coche más ligero?



¿Color más frecuente?

```
def hora_mas_avistamientos(avistamientos):
```

```
    '''
```

Devuelve la hora del día (de 0 a 23) con mayor número de avistamientos

ENTRADA:

- avistamientos: lista de tuplas con la información de los avistamientos
-> [Avistamiento(datetime, str, str, str, int, str, float, float)]

SALIDA:

- hora del día en la que se producen más avistamientos -> int

En primer lugar construiremos un diccionario cuyas claves sean las horas del día en las que se han observado avistamientos, y cuyos valores sean el número de avistamientos observados en esa hora.

Después obtendremos el máximo de los elementos del diccionario según el valor del elemento.

```
    '''
```

```
    result = dict()
```

```
    for a in avistamientos:
```

```
        clave = a.fechahora.hour
```

```
        if clave in result:
```

```
            result[clave] += 1
```

```
        else:
```

```
            result[clave] = 1
```

```
    return max(result.items(), key=lambda t: t[1])
```

TAREA

- Estudiar para el cuestionario 2