

Diccionarios: Resolución mediante doble diccionario y función auxiliar- Uso de DefaultDict

Fundamentos de Programación
Departamento de Lenguajes y Sistemas Informáticos



Diccionarios (Esquema para resolución de problemas mediante el uso de dos diccionarios y función auxiliar)

Hemos visto en la lección anterior como construir **diccionario (res)** como resultado de recorrer un **diccionario previo (aux)**, en el que los valores son listas y los valores de **res** son el resultado de alguna operación (*máximo, ordenación, mínimo, promedio, ...*) a los valores de **aux**

Ahora se trata de *que la operación adicional no es tan inmediata como las descritas, por ejemplo, una **nueva agrupación** por otro criterio.*

Supongamos que nuestro tipo Estudiantes tiene un campo más, **el grado** que cursan los estudiantes.

Estudiante=NamedTuple("estudiante", [("grado",str),("nombre",str),("edad",int),("curso",str)])

```
estudiantes=[('TI3', 'Ismael', 19, 'C4'), ('TI4', 'Ruben', 18, 'C2'), ('TI3', 'Lorena', 20, 'C2'), ('TI4', 'Rocío', 18, 'C1'), ('IS2', 'M.Mar', 19, 'C1'), ('IS3', 'David', 18, 'C3'), ('IS2', 'Mario', 20, 'C3'), ('IS2', 'Daniel', 20, 'C2'), ('IS3', 'Javier', 17, 'C1'), ('TI3', 'Daniel', 19, 'C4'), ('IS3', 'Javier', 18, 'C1'), ('TI3', 'Adrián', 18, 'C4'), ('IS2', 'Javier', 21, 'C2'), ('IS3', 'Celia', 22, 'C1'), ('TI3', 'David', 23, 'C3'), ('IS2', 'Mario', 19, 'C4'), ('IS3', 'Rocío', 18, 'C4'), ('TI3', 'Javier', 19, 'C1'), ('TI3', 'Carlos', 20, 'C2'), ('IS3', 'Guillermo', 20, 'C2'), ('IS3', 'José', 20, 'C3'), ('TI4', 'Luis', 19, 'C1'), ('TI3', 'Javier', 21, 'C4'), ('TI3', 'Fernando', 20, 'C1'), ('TI3', 'Pedro', 18, 'C3'), ('TI4', 'Ana', 20, 'C1'), ('IS2', 'Manuel', 18, 'C4')]
```

Ejercicio: Obtener un diccionario que a cada **grado** le asocie una lista de tuplas con cada curso y el promedio de edades de dicho curso

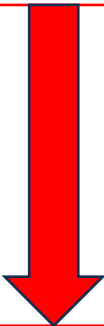


Diccionarios (Esquema para resolución de problemas mediante el uso de dos diccionarios y función auxiliar)

*Ejercicio: Obtener un diccionario que a cada **grado** le asocie una lista de tuplas con cada curso y el promedio de edades de dicho curso*

Datos de entrada

```
estudiantes=[('TI3', 'Ismael', 19, 'C4'), ('TI4', 'Ruben', 18, 'C2'), ('TI3', 'Lorena', 20, 'C2'), ('TI4', 'Rocío', 18, 'C1'), ('IS2', 'M.Mar', 19, 'C1'), ('IS3', 'David', 18, 'C3'), ('IS2', 'Mario', 20, 'C3'), ('IS2', 'Daniel', 20, 'C2'), ('IS3', 'Javier', 17, 'C1'), ('TI3', 'Daniel', 19, 'C4'), ('IS3', 'Javier', 18, 'C1'), ('TI3', 'Adrián', 18, 'C4'), ('IS2', 'Javier', 21, 'C2'), ('IS3', 'Celia', 22, 'C1'), ('TI3', 'David', 23, 'C3'), ('IS2', 'Mario', 19, 'C4'), ('IS3', 'Rocío', 18, 'C4'), ('TI3', 'Javier', 19, 'C1'), ('TI3', 'Carlos', 20, 'C2'), ('IS3', 'Guillermo', 20, 'C2'), ('IS3', 'José', 20, 'C3'), ('TI4', 'Luis', 19, 'C1'), ('TI3', 'Javier', 21, 'C4'), ('TI3', 'Fernando', 20, 'C1'), ('TI3', 'Pedro', 18, 'C3'), ('TI4', 'Ana', 20, 'C1'), ('IS2', 'Manuel', 18, 'C4')]
```

- 
- 1) un diccionario que *agrupe por grado*
 - 2) una función que *agrupe por curso* y calcule los promedios

Datos de salida

```
{ 'TI3': [ ('C4', 19.25), ('C2', 20.0), ('C3', 20.5), ('C1', 19.5) ],  
  'TI4': [ ('C2', 18.0), ('C1', 19.0) ],  
  'IS2': [ ('C1', 19.0), ('C3', 20.0), ('C2', 20.5), ('C4', 18.5) ],  
  'IS3': [ ('C3', 19.0), ('C1', 19.0), ('C4', 18.0), ('C2', 20.0) ] }
```



Diccionarios (Esquema para resolución de problemas mediante el uso de dos diccionarios y función auxiliar)

Obtener un diccionario que a cada **grado** le asocie una lista de tuplas con cada curso y el promedio de edades de dicho curso

```
aux=dict()
for e in estudiantes:
    if e.grado not in aux:
        aux[e.grado]=list()
        aux[e.grado].append(e)
res=dict()
for c,v in aux.items():
    res[c]=agrupa_por_curso(v)
return res
```

```
{'TI3': [('TI3', 'Ismael', 19, 'C4'), ('TI3', 'Lorena', 20, 'C2'), ('TI3',
    'Daniel', 19, 'C4'), ('TI3', 'Adrián', 18, 'C4'), ...],
'TI4': [('TI4', 'Ruben', 18, 'C2'), ('TI4', 'Rocío', 18, 'C1'), ('TI4',
    'Luis', 19, 'C1'), ('TI4', 'Ana', 20, 'C1')],
'IS2': [('IS2', 'M.Mar', 19, 'C1'), ('IS2', 'Mario', 20, 'C3'), ('IS2',
    'Daniel', 20, 'C2'), ('IS2', 'Javier', 21, 'C2'), ...],
'IS3': [('IS3', 'David', 18, 'C3'), ('IS3', 'Javier', 17, 'C1'), ('IS3',
    'Javier', 18, 'C1'), ('IS3', 'Celia', 22, 'C1'), ...]}
```

Por cada clave (grado), se pasará su valor (la lista). En este ejercicio ocurrirá cuatro veces.



Diccionarios (Esquema para resolución de problemas mediante el uso de dos diccionarios y función auxiliar)

Obtener un diccionario que a cada **grado** le asocie una lista de tuplas con cada curso y el promedio de edades de dicho curso

```
aux=dict()
for e in estudiantes:
    if e.grado not in aux:
        aux[e.grado]=list()
    aux[e.grado].append(e)
res=dict()
for c,v in aux.items():
    res[c]=agrupa_por_curso(v)
return res
```

Para la clave (TI3), se pasa su valor (la lista).

```
{'TI3': [('TI3', 'Ismael', 19, 'C4'), ('TI3', 'Lorena', 20, 'C2'), ('TI3', 'Daniel', 19, 'C4'), ('TI3', 'Adrián', 18, 'C4'), ...]}
```

```
def agrupa_por_curso(est_de_un_grado:List[Estudiante])>List[Tuple[str,float]]:
```

```
    aux=dict()
    for e in est_de_un_grado:
        if e.curso not in aux:
            aux[e.curso]=list()
        aux[e.curso].append(e.edad)
```

```
{'C4': [19, 19, 18, 21], 'C2': [20, 20], 'C3': [23, 18], 'C1': [19, 20]}
```

```
    res=dict()
    for c,v in aux.items():
        res[c]=sum(v)/len(v)
    return list(res.items())
```

```
[('C4', 19.25), ('C2', 20.0), ('C3', 20.5), ('C1', 19.5)]
```



Diccionarios (Esquema para resolución de problemas mediante el uso de dos diccionarios y función auxiliar)

Obtener un diccionario que a cada **grado** le asocie una lista de tuplas con cada curso y el promedio de edades de dicho curso

```
aux=dict()
for e in estudiantes:
    if e.grado not in aux:
        aux[e.grado]=list()
        aux[e.grado].append(e)
res=dict()
for c,v in aux.items():
    res[c]=agrupa_por_curso(v)
return res
```

Para la clave (TI4), se pasa su valor (la lista)

```
{
  'TI4': [ ('TI4', 'Ruben', 18, 'C2'), ('TI4', 'Rocío', 18, 'C1'), ('TI4',
    'Luis', 19, 'C1'), ('TI4', 'Ana', 20, 'C1') ],
```

```
def agrupa_por_curso(est_de_un_grado:List[Estudiante])->List[Tuple[str,float]]:
```

```
    aux=dict()
    for e in est_de_un_grado:
        if e.curso not in aux:
            aux[e.curso]=list()
            aux[e.curso].append(e.edad)
```

```
{ 'C2': [18], 'C1': [18,19,20] }
```

```
    res=dict()
    for c,v in aux.items():
        res[c]=sum(v)/len(v)
    return list(res.items())
```

```
[ ('C2', 18.0), ('C1', 19.0) ]
```



Diccionarios (Esquema para resolución de problemas mediante el uso de dos diccionarios y función auxiliar)

Obtener un diccionario que a cada **grado** le asocie una lista de tuplas con cada curso y el promedio de edades de dicho curso

```
aux=dict()
for e in estudiantes:
    if e.grado not in aux:
        aux[e.grado]=list()
        aux[e.grado].append(e)
res=dict()
for c,v in aux.items():
    res[c]=agrupa_por_curso(v)
return res
```

Para la clave (IS2), se pasa su valor (la lista).

```
'IS2': [('IS2', 'M.Mar', 19, 'C1'), ('IS2', 'Mario', 20, 'C3'), ('IS2', 'Daniel', 20, 'C2'), ('IS2', 'Javier', 21, 'C2'), ...]
```

```
def agrupa_por_curso(est_de_un_grado:List[Estudiante])>List[Tuple[str,float]]:
```

```
    aux=dict()
    for e in est_de_un_grado:
        if e.curso not in aux:
            aux[e.curso]=list()
            aux[e.curso].append(e.edad)
```

```
{'C1': [19], 'C3': [20], 'C2': [20, 21], 'C4': [19, 18]}
```

```
    res=dict()
    for c,v in aux.items():
        res[c]=sum(v)/len(v)
    return list(res.items())
```

```
[('C1', 19.0), ('C3', 20.0), ('C2', 20.5), ('C4', 18.5)]
```



Diccionarios (Esquema para resolución de problemas mediante el uso de dos diccionarios y función auxiliar)

Obtener un diccionario que a cada **grado** le asocie una lista de tuplas con cada curso y el promedio de edades de dicho curso

```
aux=dict()
for e in estudiantes:
    if e.grado not in aux:
        aux[e.grado]=list()
        aux[e.grado].append(e)
res=dict()
for c,v in aux.items():
    res[c]=agrupa_por_curso(v)
return res
```

```
{
    'IS3': [ ('IS3', 'David', 18, 'C3'), ('IS3', 'Javier', 17, 'C1'), ('IS3', 'Javier', 18, 'C1'), ('IS3', 'Celia', 22, 'C1'), ... ] }
```

Para la clave (TI3), se pasa su valor (la lista).

```
def agrupa_por_curso(est_de_un_grado:List[Estudiante])->List[Tuple[str,float]]:
```

```
    aux=dict()
    for e in est_de_un_grado:
        if e.curso not in aux:
            aux[e.curso]=list()
            aux[e.curso].append(e.edad)
```

```
{ 'C3': [18,20], 'C1': [17,18,22], 'C4': [18], 'C2': [20] }
```

```
    res=dict()
    for c,v in aux.items():
        res[c]=sum(v)/len(v)
    return list(res.items())
```

```
[ ('C3', 19.0), ('C1', 19.0), ('C4', 18.0), ('C2', 20.0) ]
```




Diccionarios (Esquema para resolución de problemas mediante el uso de dos diccionarios y función auxiliar)

Obtener un diccionario que a cada grado le asocie una lista de tuplas con cada curso y el promedio de edades de dicho curso

```
aux=dict()
for e in estudiantes:
    if e.grado not in aux:
        aux[e.grado]=list()
        aux[e.grado].append(e)
res=dict()
for c,v in aux.items():
    res[c]=agrupa_por_curso(v)
return res
```

```
{'TI3': [('TI3', 'Ismael', 19, 'C4'), ('TI3', 'Lorena', 20, 'C2'), ('TI3',
    'Daniel', 19, 'C4'), ('TI3', 'Adrián', 18, 'C4'), ...],
'TI4': [('TI4', 'Ruben', 18, 'C2'), ('TI4', 'Rocío', 18, 'C1'), ('TI4',
    'Luis', 19, 'C1'), ('TI4', 'Ana', 20, 'C1')],
'IS2': [('IS2', 'M.Mar', 19, 'C1'), ('IS2', 'Mario', 20, 'C3'), ('IS2',
    'Daniel', 20, 'C2'), ('IS2', 'Javier', 21, 'C2'), ...],
'IS3': [('IS3', 'David', 18, 'C3'), ('IS3', 'Javier', 17, 'C1'), ('IS3',
    'Javier', 18, 'C1'), ('IS3', 'Celia', 22, 'C1'), ...]}
```

```
{'TI3': [('C4', 19.25), ('C2', 20.0), ('C3', 20.5), ('C1', 19.5)],
'TI4': [('C2', 18.0), ('C1', 19.0)],
'IS2': [('C1', 19.0), ('C3', 20.0), ('C2', 20.5), ('C4', 18.5)],
'IS3': [('C3', 19.0), ('C1', 19.0), ('C4', 18.0), ('C2', 20.0)]}
```



Diccionarios: (Construcción con defaultdict)

Para construir diccionarios se han visto dos esquemas:

Por ejemplo, para contar cuantos alumnos hay en cada grado

```
res=dict()
for e in estudiantes:
    if e.grado not in res:
        res[e.grado]=0
    res[e.grado]+=1
```

```
res=dict()
for e in estudiantes:
    if e.grado not in res:
        res[e.grado]=1
    else:
        res[e.grado]+=1
```

Pero con el uso de **DefaultDict** no es necesario comprobar si existe o no la clave. Únicamente hay que *instanciar* adecuadamente **DefaultDict**:

```
from typing import defaultdict
res=defaultdict(int)
for e in estudiantes:
    res[e.grado]+=1
```

```
print(res)           → defaultdict(<class 'int'>, {'TI3':10,'TI4':4,'IS2':6,'IS3':7})
print(res.items())   → dict_items([('TI3',10),('TI4',4),('IS2',6),('IS3',7)])
print(list(res.items())) → [('TI3',10),('TI4',4),('IS2',6),('IS3',7)]
```



Diccionarios: (Construcción con DefaultDict)

Para construir diccionarios de listas o conjuntos se han visto dos esquemas:

Por ejemplo, los distintos nombres de cursos por grados

```
res=dict()
for e in estudiantes:
    if e.grado not in res:
        res[e.grado]=set()
    res[e.grado].add(e.curso)
```

```
res=dict()
for e in estudiantes:
    if e.grado not in res:
        res[e.grado]=set()
        res[e.grado].add(e.curso)
    else:
        res[e.grado].add(e.curso)
```

DefaultDict se
instancia a *set*

```
from typing import DefaultDict
res=DefaultDict(set)
for e in estudiantes:
    res[e.grado].add(e.curso)
```

```
print(res) → defaultdict(<class 'set'>, {'TI3': {'C1', 'C2', 'C3', 'C4'}, 'TI4': {'C2', 'C1'}, 'IS2': {'C2', 'C3', 'C1', 'C4'}, 'IS3': {'C2', 'C3', 'C1', 'C4'}})
print(res.items()) → dict_items([('TI3', {'C1', 'C2', 'C3', 'C4'}), ('TI4', {'C2', 'C1'}), ('IS2', {'C2', 'C3', 'C1', 'C4'}), ('IS3', {'C2', 'C3', 'C1', 'C4'})])
print(list(res.items())) → [('TI3', {'C1', 'C2', 'C3', 'C4'}), ('TI4', {'C2', 'C1'}), ('IS2', {'C2', 'C3', 'C1', 'C4'}), ('IS3', {'C2', 'C3', 'C1', 'C4'})]
```



Ejercicio:

Proyecto T10_TMBD_Movie:

- Descargar e Implementar el proyecto