



El futuro digital
es de todos

MinTIC

Hechos

QUE

CONECTAN



Taller Diccionario Conjuntos

Presentada por el equipo de tutores

Universidad
Industrial de
Santander



Mision
TIC 2022

Conjuntos

Un conjunto es una colección no ordenada de objetos únicos. Python provee este tipo de datos «por defecto» al igual que otras colecciones más convencionales como las lista, tuplas y diccionarios.

Los conjuntos son inmutables

Creación de conjuntos en python

Para crear un conjuntos colocamos nombre de la variables abrimos y cerramos {}.

Ejemplo:

Código:

```
colores = {"Rojo", "Amarillo", "Azul", "Rosado", "Azul"}  
print(type(colores))  
print(colores)
```

Terminal

<class 'set'>

{'Rojo', 'Amarillo', 'Rosado', 'Azul'}

Creación de conjuntos en python

```
letras = set("Holamundo")  
print(type(letras))  
print(letras)
```

```
rango = set(range(20,1000,100))  
print(type(rango ))  
print(rango)
```

```
unicos = set([3, 5, 6, 1, 5])  
print(type(unicos))  
print(unicos)
```

Creación de conjuntos en python

```
conjunto2 = set([(1, 2, 3), (4, 5), (6, 7, 8, 9)])  
print(type(conjunto2))  
print(conjunto2)
```

No es posible crear conjuntos Los elementos de un conjunto deben ser inmutables

```
s = {[2, 4], [6, 1]}
```

Las tuplas son inmutables si se puede crear

```
h = {(2, 5), (9, 1)}  
print(type(h))  
print(h)
```

¿Acceder elementos de un conjuntos?

No se puede acceder a elementos de un conjuntos, porque estos no son ordenados.

```
conjunto3 = {'a', 'b', 'c'}  
print(conjunto3[0])
```

Podemos por medio de un ciclo for

```
for i in conjunto3:  
    print(i)
```

Operaciones entre conjuntos

Mostrar tamaño un conjunto

```
conjuntofruta = {"Pera", "Mango", "Banano"}  
tam = len(conjuntofruta)  
print(tam)
```

Si un elemento esta en el conjunto:

```
conjuntounumero = {1,2,3,4,8}  
print(9 in conjunto3)  
print(3 in conjuntounumero)  
print(10 not in conjuntounumero)
```

Operaciones entre conjuntos

$a = \{1, 2, 3, 4\}$

$b = \{2, 4, 6, 8\}$

$\text{interseccion} = a \cap b$

$\text{union} = a \cup b$

$\text{diferencia} = a - b$ #Elemento de a que no están en b

$\text{diferenciasimetrica} = b \Delta a$ #Diferencia simétrica

$\{2, 4\}$

$\{1, 2, 3, 4, 6, 8\}$

$\{1, 3\}$

$\{1, 3, 6, 8\}$

Operaciones entre conjuntos

Subconjuntos

$f = \{1, 2\}$

$a = \{1, 2, 3, 4\}$

$\text{subconjunto} = f < a$

True

$f = \{1, 2, 3, 4\}$

$a = \{1, 2, 3, 4\}$

$\text{subconjunto} = f \leq a$

$\text{subconjuntoa} = f < a$

True

False

Añadir y remover elementos de un conjunto

Añadir:

```
conjuntox = {6, 1, 5, 4, 3}
conjuntox.add(-37)
print(conjuntox)
{1, 3, 4, 5, 6, -37}
```

Eliminar elementos:

```
conjuntox.remove(5)
print(conjuntox)
{1, 3, 4, 6, -37}
```

Ejercicio

Solicitar al usuario que ingrese los nombres de los alumnos de nivel primario de una escuela, finalizando al ingresar “x”, A continuación, solicitar que ingrese los nombres de los alumnos de nivel secundario, finalizando al ingresar “x”, informar los nombres de **todos los alumnos** de nivel primario y los nivel secundario sin repeticiones informar que **nombres se repiten** entre nivel primario y secundario informar que **nombres no se repiten** entre primera y el secundario.

Ejercicio

Mostrar la **diferencia** entre conjuntos de colores.

conjuntoa = {"Rojo", "Amarillo", "Verde", "Blanco"}

conjuntob = {"Blanco", "Azul", "Rosado", "Amarillo", "Gris"}

Diccionario

Un **diccionario** es un tipo de datos que sirve para asociar pares de objetos.

Un diccionario puede ser visto como una colección de **llaves**, cada una de las cuales tiene asociada un **valor**. Las llaves no están ordenadas y no hay llaves repetidas. La única manera de acceder a un valor es a través de su llave.

Las llaves deben ser un tipo dato inmutables es decir lista no pueden ser

Creación de Diccionario

Los diccionarios se crean usando llaves ({ y }). La llave y el valor van separados por dos puntos

Ejemplo:

Diccionario de teléfonos

```
telefonos =  
{ "Pablo": 3142354675, "Miguel": 3212354675, "Ronaldo": 3162354675, "Alfonso": 3132354675,  
}
```

Diccionario vacío

```
a = dict()  
d = {}  
print(type(a))  
print(type(d))
```

Buscar valor especifico

Buscamos por la llave

```
print(telefonos["Pablo"])  
print(telefonos["Alfonso"])
```

3142354675

3132354675

¿Si la llave no esta en el diccionario?

```
print(telefonos["Manuel"])
```

```
print(telefonos["Manuel"])
```

KeyError: 'Manuel'

Agregar nuevos llaves y valores

```
telefonos["Teresa"] = 3204567312  
print(telefonos)
```

```
{'Pablo': 3142354675, 'Miguel': 3212354675, 'Ronaldo': 3162354675, 'Alfonso':  
3132354675, 'Teresa': 3204567312}
```

¿Sí se agregar un valor a una llave que estaba?

```
telefonos["Ronaldo"] = 3172345631  
print(telefonos)
```

```
{'Pablo': 3142354675, 'Miguel': 3212354675, 'Ronaldo': 3172345631, 'Alfonso':  
3132354675, 'Teresa': 3204567312}
```


Eliminar elementos diccionario

```
del telefonos["Pablo"]  
print(telefonos)
```

```
{'Miguel': 3212354675, 'Ronaldo': 3172345631, 'Alfonso': 3132354675, 'Teresa':  
3204567312}
```

Ciclo for y diccionario

```
for i in telefonos:  
    print(i)
```

Mostrara todas las llaves del diccionario

```
for i in telefonos.values():  
    print(i)
```

Mostrara todas las valores del diccionario

Ciclo for y diccionario

```
for k, v in telefonos.items():  
    print("El telefono de ",k," es: ",v)
```

Mostrara tanto las llaves como los valores

Crear de un lista un diccionario

```
key_list = ['name', 'age', 'address']  
value_list = ['Johnny', '27', 'New York']  
dict_from_list = dict(zip(key_list, value_list))  
print(dict_from_list)
```

{'name': 'Johnny', 'age': '27', 'address': 'New York'}

Crear de un diccionario una lista

```
telefonos =  
{ "Pablo": 3142354675, "Miguel": 3212354675, "Ronaldo": 3162354675, "Alfonso": 3132354675,  
}  
keylist = list(telefonos)  
valuelist = list(telefonos.values())
```

['Pablo', 'Miguel', 'Ronaldo', 'Alfonso']

[3142354675, 3212354675, 3162354675, 3132354675]

Tamaño de un diccionario

```
telefonos =  
{ "Pablo":3142354675, "Miguel":3212354675, "Ronaldo":3162354675, "Alfonso":3132354675,  
}  
tam = len(telefonos)  
print(tam)
```

4

Un elemento está o no esta en un diccionario

```
patas = {'gato': 4, 'humano': 2, 'pulpo': 8, 'perro': 4, 'ciempies': 100}
```

```
print(perro in patas)
```

```
print(gusano in patas)
```

True

False

```
print('gusano' not in patas)
```

True

Archivos JSON

JavaScript Object Notation es un formato ligero de intercambio de datos. Leerlo y escribirlo es simple para humanos, mientras que para las máquinas es simple interpretarlo y generarlo. JSON es un formato de texto que es completamente independiente del lenguaje pero utiliza convenciones que son ampliamente conocidos por los programadores de la familia de lenguajes. Estas propiedades hacen que JSON sea un lenguaje ideal para el intercambio de datos

JSON está constituido por dos estructuras:

Una colección de pares de nombre/valor. En varios lenguajes esto es conocido como un *objeto*, registro, estructura, diccionario, tabla hash, lista de claves o un arreglo asociativo.

Para utilizar archivos JSON en python debemos importar el modulo json de la siguiente manera

```
import json
```


Generar archivo json

```
import json
data = {}
data['clients'] = []
data['clients'].append({
    'first_name': 'Theodoric',
    'last_name': 'Rivers',
    'age': 36,
    'amount': 1.11})

with open('data.json', 'w') as file:
    json.dump(data, file, indent=4)
```



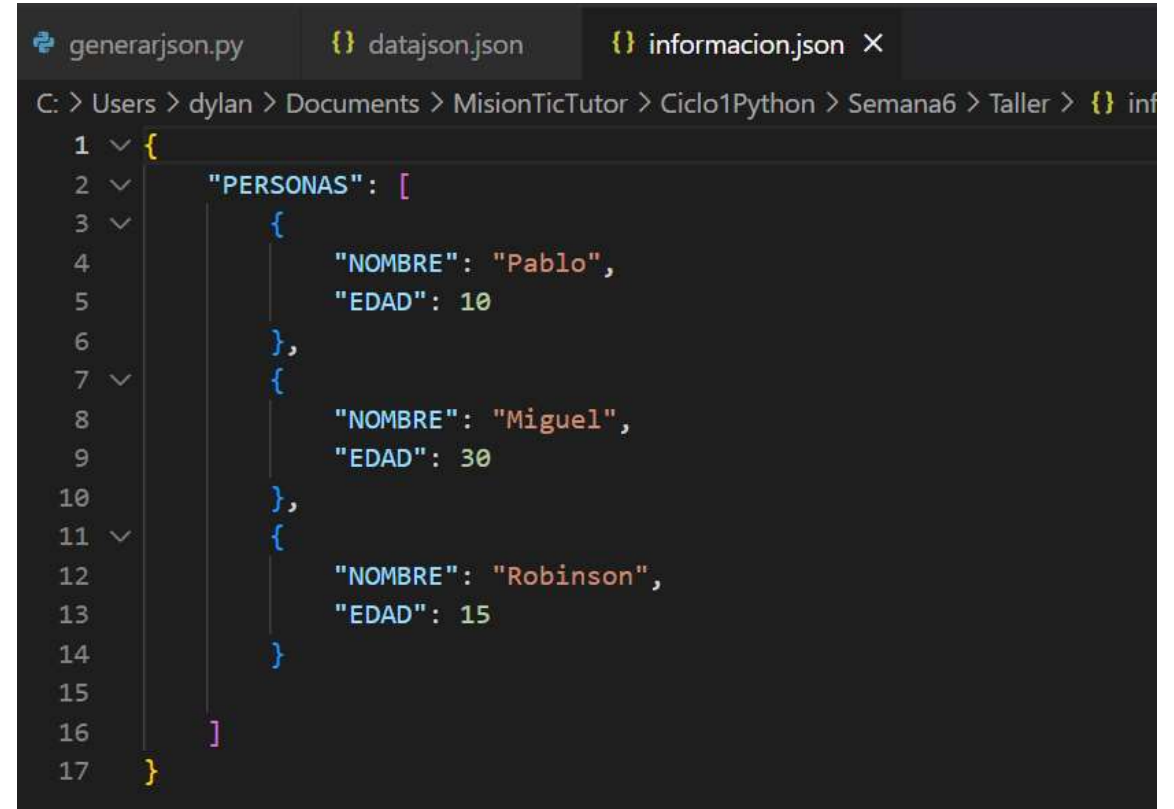
The screenshot shows a code editor with three tabs: 'generarjson.py', 'datajson.json', and 'data.json'. The 'datajson.json' tab is active, displaying the following JSON content:

```
1  {
2      "clients": [
3          {
4              "first_name": "Theodoric",
5              "last_name": "Rivers",
6              "age": 36,
7              "amount": 1.11
8          }
9      ]
10 }
```

Abrir un archivo JSON en Python

```
import json
with open('informacion.json') as file:
    data = json.load(file)
print(data)
```

```
{'PERSONAS': [{'NOMBRE': 'Pablo', 'EDAD': 10}, {'NOMBRE': 'Miguel', 'EDAD': 30}, {'NOMBRE': 'Robinson', 'EDAD': 15}]}
```



```
generarjson.py  datajson.json  informacion.json X
C: > Users > dylan > Documents > MisionTicTutor > Ciclo1Python > Semana6 > Taller > {} inf
1  {
2      "PERSONAS": [
3          {
4              "NOMBRE": "Pablo",
5              "EDAD": 10
6          },
7          {
8              "NOMBRE": "Miguel",
9              "EDAD": 30
10         },
11         {
12             "NOMBRE": "Robinson",
13             "EDAD": 15
14         }
15     ]
16 }
17 }
```

Ejercicio

Escribir un programa que guarde en una variable el diccionario {'Euro':'€', 'Dollar':'\$', 'Yen':'¥'}, pregunte al usuario por una divisa y muestre su símbolo o un mensaje de aviso si la divisa no está en el diccionario

Escribir un programa que guarde en un diccionario los precios de las frutas de la tabla, pregunte al usuario por una fruta, un número de kilos y muestre por pantalla el precio de ese número de kilos de fruta. Si la fruta no está en el diccionario debe mostrar un mensaje informando de ello.

Escribir un programa que almacene el diccionario con los créditos de las asignaturas de un curso {'Matemáticas': 6, 'Física': 4, 'Química': 5} y después muestre por pantalla los créditos de cada asignatura en el formato <asignatura> tiene <créditos> créditos, donde <asignatura> es cada una de las asignaturas del curso, y <créditos> son sus créditos. Al final debe mostrar también el número total de créditos del curso.

Ejercicio

Escribir un programa que cree un diccionario simulando una cesta de la compra. El programa debe preguntar el artículo y su precio y añadir el par al diccionario, hasta que el usuario decida terminar. Después se debe mostrar por pantalla la lista de la compra y el coste total, con el siguiente formato.

Calcular el promedio de nota final de un curso, y previamente de la cada alumnos de un archivo json llamado notasalumno.json