Autores: María Inmaculada Rodríguez García, María Gema Carrasco García, Javier González Enrique, Juan Jesús Ruiz Aguilar, Ignacio J. Turias Domínguez. Universidad de Cádiz

Módulo 2. Manipulación de datos
2.1. Series en Python y tablas en MATLAB

- **Librería Pandas para Python.**

Pandas. Es una librería de Python especializada en la manipulación y el análisis de datos, por tanto, cuenta con funciones para el análisis, la limpieza, la exploración y la manipulación de datos.

Proporciona estructuras de datos de alto nivel y herramientas para tablas (DataFrame) y series (Series).

Construida sobre *numpy*.

Para obtener una nueva serie de booleanos se usa: **serie.isnull(), serie.notnull()**. Se puede usar como filtro: **serie[serie.isnull()], serie[serie.notnull()].**

Trabajaremos con series usando Pandas.

- Una serie es un *array* de una dimensión con un índice.
- Este índice puede ser una etiqueta para caracterizar mejor los elementos guardados.
- Permite almacenar información de cualquier tipo (cadenas de texto, flotantes o enteros, entre otros)

Documentación: Documentación general sobre Pandas

Series en Python con Pandas

- En **MATLAB** el equivalente a series es simular mediante tablas con al menos dos columnas; una de etiquetas y otra de valores. No es necesario instalar ningún Toolbox adicional, es decir, se manejan estructuras de datos nativas.

Ejercicio 62_01. Creación de una serie desde numpy con pandas en Python

```
# Importamos las librerías necesarias
import numpy as np
import pandas as pd

# Primero creamos un array de numpy con los datos
v = np.array([100, 200, 300],dtype = np.int32)
# Luego una lista con los datos de las etiquetas
filas = ["2018", "2019", "2020"]
# Creamos la serie
serie1 = pd.Series(v, index = filas)

# Mostramos
print(serie1)
```

```
    2018    100
    2019    200
    2020    300
dtype: int32
```

Autores: María Inmaculada Rodríguez García, María Gema Carrasco García, Javier González Enrique, Juan Jesús Ruiz Aguilar, Ignacio J. Turias Domínguez. Universidad de Cádiz

Ejercicio 62_02. Creación de una tabla en MATLAB

```
% Primero creamos un array de MATLAB con los datos
v = [100, 200, 300];
% Luego una lista con los datos de las etiquetas
filas = {'2018', '2019', '2020'};
% Creamos la tabla
tabla1 = table(v', 'RowNames', filas);
% Mostramos
disp('Tabla 1:');
disp(tabla1);
```

```
Tabla 1:
            Var1
            ____
    2018    100
    2019    200
    2020    300
```

Ejercicio 63_01. Definición de forma directa desde Numpy con Pandas en Python

```
# Los datos de una serie de Pandas son arrays de Numpy
nat = pd.series([10.7, 7.5, 7.1, 17.8, 7.9], index = ["Murcia", "Cantabria",
"Galicia", "Melilla", "Canarias”, dtype = float)
name = "Tasa de Natalidad"]
# Mostramos
print(nat)
```

```
    Murcia             10.7
    Cantabria           7.5
    Galicia             7.1
    Melilla            17.8
    Canarias            7.9
Name: Tasa de Natalidad, dtype: float64
```

Ejercicio 63_02. Definición de forma directa en Matlab

```
nat = [10.7, 7.5, 7.1, 17.8, 7.9];
index_nat = {'Murcia', 'Cantabria', 'Galicia', 'Melilla', 'Canarias'};
tabla2 = table(nat', 'RowNames', index_nat, 'VariableNames', {'Tasa_de_Natalidad'});
% Mostramos
disp('Tabla 2:');
disp(tabla2);
```

```
Tabla 2:
                Tasa_de_Natalidad
                _________________
    Murcia            10.7
    Cantabria          7.5
    Galicia            7.1
    Melilla           17.8
    Canarias           7.9
```

Autores: María Inmaculada Rodríguez García, María Gema Carrasco García, Javier González Enrique, Juan Jesús Ruiz Aguilar, Ignacio J. Turias Domínguez. Universidad de Cádiz

Ejercicio 64_01. Definición sin especificar etiquetas

```
a = [4, 8, 16, 32, 64]

# se crea un índice por defecto empezando en el 0
serie2 = pd.Series(a)
# Mostramos
print(serie2)
```

```
 0   4
 1   8
 2   16
 3   32
 4   64
```

Ejercicio 64_02. Definición sin especificar etiquetas

```
a = [4, 8, 16, 32, 64];
tabla3 = table(a', 'VariableNames', {'Columna_A'});
% Mostramos
disp('Tabla 3:');
disp(tabla3);
```

```
Tabla 3:
    Columna_A
    _________

        4
        8
       16
       32
       64
```

Ejercicio 65_01. Creación de una serie desde diccionario

```
# Las claves del diccionario pasarán a ser las etiquetas
# Definimos diccionario
emisiones_co2 = {"Enero": 12, "Febrero": 28, "Marzo": 40, "Abril": 10}
series3 = pd.Series(emisiones_co2)
print(series3)
```

```
    Enero           12
    Febrero         28
    Marzo           40
    Abril           10
```

Ejercicio 65_02. Creación de una tabla desde diccionario

```
% Definimos diccionario
emisiones_co2 = {'Enero', 12; 'Febrero', 28; 'Marzo', 40; 'Abril', 10};
```

```
tabla4 = table(emisiones_co2(:, 2), 'RowNames', emisiones_co2(:, 1), 'VariableNames',
{'Emisiones_CO_{2}'});
disp('Tabla 4:');
disp(tabla4);

Tabla 4:
               Emisiones_CO_{2}
               ________________

    Enero           {[12]}
    Febrero         {[28]}
    Marzo           {[40]}
    Abril           {[10]}
```

```
Ejercicio 66_01. Creación de una serie usando únicamente las claves que nos interesen

series4 = pd.Series(emisiones_co2, index = ["Febrero", "Abril"])
print(series4)

    Febrero       28
    Abril         10
```

```
Ejercicio 66_02. Creación de una tabla usando únicamente las claves que nos interesen

% Podemos crear una tabla usando únicamente las claves que nos interesen:
indices_interes = {'Febrero', 'Abril'};
tabla5 = table(emisiones_co2(ismember(emisiones_co2(:, 1), indices_interes), 2),
'RowNames', indices_interes, 'VariableNames', {'Emisiones_CO2'});
disp('Tabla 5:');
disp(tabla5);

Tabla 5:
               Emisiones_CO2
               _____________

    Febrero       {[28]}
    Abril         {[10]}
```

```
Ejercicio 67_01. Acceso a los datos de la serie

# Se puede acceder por el índice o por la etiqueta
print("Imprimimos primer elemento de la serie: ")
print(nat[0]) # Acceso por el índice
print("Imprimimos el valor de la serie para Cantabria")
print(nat["Cantabria"]) # Acceso por la etiqueta


Imprimimos primer elemento de la serie:
   10.7000
Imprimimos el valor de la serie para Cantabria:
    7.5000
```

Ejercicio 67_02. Acceso a los datos de la tabla

```
% Acceso a los datos de la tabla
disp('Imprimimos primer elemento de la tabla 2:');
disp(tabla2{'Murcia', 'Tasa_de_Natalidad'});
disp('Imprimimos el valor de la tabla 2 para Cantabria:');
disp(tabla2{'Cantabria', 'Tasa_de_Natalidad'});
```

```
Imprimimos primer elemento de la tabla 2:
   10.7000
Imprimimos el valor de la tabla 2 para Cantabria:
    7.5000
```

Ejercicio 68_01. Datos vacíos en una serie en Python

```
serie1 = pd.Series([1, 2, 3, np.NaN, 5])
print("Imprimimos la serie")
print(serie1)
print("Imprimimos resultado isna()")
print(serie1.isna())
print("Imprimimos resultado isnull()")
print(serie1.isnull())
print("Filtramos los no nulos")
print(serie1[serie1.notnull()])
print("Filtramos mostrando los nulos")
print(serie1[serie1.isnull()])

print(serie1)
0    1.0
1    2.0
2    3.0
3    NaN
4    5.0
dtype: float64

print(serie1.isna())
0    False
1    False
2    False
3     True
4    False
dtype: bool

print(serie1.isnull())
0    False
1    False
2    False
3     True
4    False
dtype: bool
```

Autores: María Inmaculada Rodríguez García, María Gema Carrasco García, Javier González Enrique, Juan Jesús Ruiz Aguilar, Ignacio J. Turias Domínguez. Universidad de Cádiz

```
print(serie1[serie1.isnull()])
3   NaN
dtype: float64
```

Ejercicio 68_01. Datos vacíos en una tabla en MATLAB

```
Names = {'USA'; 'España'; 'Italia'; 'Argentina'; 'Francia'};
Numbers = [1; 2; 3; NaN; 5];
T = table(Names, Numbers)

T = 5×2 table

        Names         Numbers
    ______________    _______

    {'USA'      }        1
    {'España'   }        2
    {'Italia'   }        3
    {'Argentina'}      NaN
    {'Francia'  }        5


% Filtrar nulos y no nulosT.Names(~isnan(T.Numbers))
disp('Filtramos los no nulos:');
disp(T.Names(~isnan(T.Numbers)));

disp('Filtramos mostrando los nulos:');
disp(T.Numbers(isnan(T.Numbers)));
```