Introducción a Pandas

Pandas es una biblioteca que proporciona estructuras de datos y herramientas de análisis de datos de alto rendimiento y faciles de usar.

- La estructura de datos principal es el DataFrame, que puede considerarse como una tabla 2D en memoria (como una hoja de cálculo, nombres de columna y etiquetas de fila).
- Muchas funciones disponbles en excel estan disponibles mediante programacion, como crear las talas dinámicas, calcular columnas basadas en otras columnas, trazar gráficos, etc.
- Proporciona un alto rendimiento para manipular(unir, dividir, modificar,etc.) grandes volumenes de datos.

Import

In [1]:
```
import pandas as pd
```

Estructuras de datos en Pandas

La biblioteca Pandas, de manera genérica, contiene las siguientes estructuras de datos:

- **Series**: Array de una dimensión.
- **DataFrame**: Se corresponde con una tabla de dos dimensiones.
- **Panel**: Similar aun diccionario de DataFrames.

Creación del Objeto Series

In [2]:
```
# Creación del objeto Series.
s = pd.Series([2, 4, 6, 8, 10])
print(s)
```

Creación del Objeto Series

In [2]:
```
# Creación del objeto Series.
s = pd.Series([2, 4, 6, 8, 10])
print(s)
```

```
0     2
1     4
2     6
3     8
4    10
dtype: int64
```

In [3]:
```
# Creación de un objeto series e inicializarlo con un diccionario de Python.
Altura = {"Emilio": 169, "Anel": 145, "Chucho": 170, "Jocelin": 170}
s = pd.Series(Altura)
print(s)
```

```
Emilio     169
Anel       145
Chucho     170
Jocelin    170
dtype: int64
```

In [4]:
```
# Creación de un Objeto Series e inicializarlo con algunos elemntos de un diccionario de Python.
Altura = {"Emilio": 169, "Anel": 145, "Chucho": 170, "Jocelin": 170}
s = pd.Series(Altura, index = ["Jocelin", "Emilio"])
print(s)
```

```
Jocelin    170
Emilio     169
dtype: int64
```

In [5]:
```
# Creación de un objeto Series e inicializarlo con un escalar.
s = pd.Series(34, ["Num1", "Num2", "Num3", "Num4"])
print(s)
```

```
Num1    34
Num2    34
Num3    34
Num4    34
dtype: int64
```

Acceso a los elementos de un array

Cada elemento es un objeto Series tiene un identificador que se denomina **index label** .

In [7]:
```
# Crear un Objeto Series
s = pd.Series([2, 4, 6, 8], index=["Num1", "Num2", "Num3", "Num4"])
print(s)
```

```
Num1    2
Num2    4
Num3    6
Num4    8
dtype: int64
```

In [8]:
```
## Acceder al tercer elemento del objeto
s["Num3"]
```

Out[8]: np.int64(6)

In [9]:
```
# Tambien se puede acceder por posición.
s[2]
```

```
/tmp/ipykernel_6471/2186343123.py:2: FutureWarning: Series.__getitem__ treating keys as positions is deprecated. In
a future version, integer keys will always be treated as labels (consistent with DataFrame behavior). To access a va
lue by position, use `ser.iloc[pos]`
  s[2]
```

Out[9]: np.int64(6)

In [11]:
```
# loc es la forma estandar de acceder a un elemento de un Objeto Series por atributo.
s.loc["Num3"]
```

Out[11]: np.int64(6)

In [12]:
```
# iloc es la forma estandar de acceder a un elemento de un Objeto Series por posición
s.iloc[2]
```

Out[12]: np.int64(6)

In [13]:
```
# Accedendo al tercer elemento por posición.
s.iloc[2:4]
```

Out[13]:
```
Num3    6
Num4    8
dtype: int64
```

Out[11]: np.int64(6)

In [12]:
```
# iloc es la forma estandar de acceder a un elemento de un Objeto Series por posición
s.iloc[2]
```

Out[12]: np.int64(6)

In [13]:
```
# Accedendo al tercer elemento por posición.
s.iloc[2:4]
```

Out[13]:
```
Num3    6
Num4    8
dtype: int64
```

Operaciones aritméticas con Series

In [14]:
```
# Crear un objeto Series
s = pd.Series([2, 4, 6, 8, 10])
print(s)
```

```
0     2
1     4
2     6
3     8
4    10
dtype: int64
```

In [15]:
```
# Los obejtos series son similares y compatibles con los Arrays de Numpy.
import numpy as np
# ufun de Numpy para sumar los elementos.
np.sum(s)
```

Out[15]: np.int64(30)

In [16]:
```
s *2
```

Out[16]:
```
0     4
1     8
2    12
3    16
4    20
```

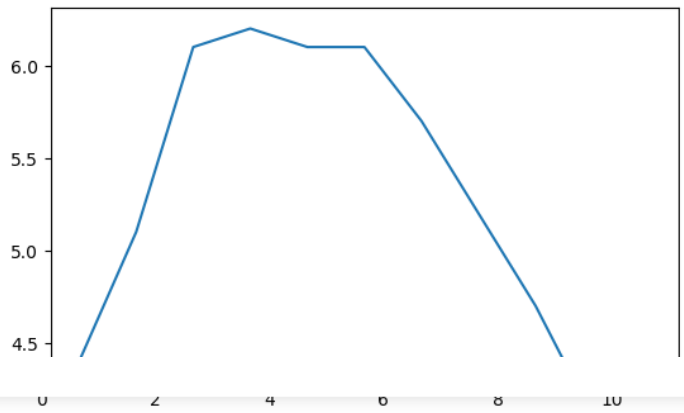
Representación gráfica de un objeto Series

In [19]:
```
# Crear un objeto Series denominado Temperaturas.
Temperaturas = [4.4, 5.1, 6.1, 6.2, 6.1, 6.1, 5.7, 5.2, 4.7, 4.1, 3.9]
s = pd.Series(Temperaturas, name="Temperaturas")
s
```

Out[19]:
```
0     4.4
1     5.1
2     6.1
3     6.2
4     6.1
5     6.1
6     5.7
7     5.2
8     4.7
9     4.1
10    3.9
Name: Temperaturas, dtype: float64
```

In [20]:
```
# Representación gráfica del objeto Series.
%matplotlib inline
import matplotlib.pyplot as plt

s.plot()
plt.show()
```



Creación de un objeto DataFrame.

In [22]:
```
# Creación de un DataFrame e inicializarlo  con un diccionario de objetos Series.
Personas = {
    "Peso": pd.Series([72, 60, 74, 73], ["Emilio", "Anel", "Chucho", "Jocelin"]),
    "Altura": pd.Series({"Emilio": 169, "Anel": 145, "Chucho": 170, "Jocelin": 170}),
    "Mascotas": pd.Series([2, 9], ["Anel", "Jocelin"])
}

df = pd.DataFrame(Personas)
df
```

Out[22]:

	Peso	Altura	Mascotas
Anel	60	145	2.0
Chucho	74	170	NaN
Emilio	72	169	NaN
Jocelin	73	170	9.0

Es posible forzar el DataFrame a que presente determinadas columnas y en orden determinado.

In [24]:
```
# Creación de un DataFrame e inicializarlo con un diccionario de objeto de series.
Personas = {
    "Peso": pd.Series([72, 60, 74, 73], ["Emilio", "Anel", "Chucho", "Jocelin"]),
    "Altura": pd.Series({"Emilio": 169, "Anel": 145, "Chucho": 170, "Jocelin": 170}),
    "Mascotas": pd.Series([2, 9], ["Anel", "Jocelin"])
}

df = pd.DataFrame(
    Personas,
    columns = ["Altura", "Peso"],
    index = ["Chucho", "Emilio"])
df
```

Out[24]:

	Altura	Peso

Creación de un objeto DataFrame.

In [22]:
```
# Creación de un DataFrame e inicializarlo  con un diccionario de objetos Series.
Personas = {
    "Peso": pd.Series([72, 60, 74, 73], ["Emilio", "Anel", "Chucho", "Jocelin"]),
    "Altura": pd.Series({"Emilio": 169, "Anel": 145, "Chucho": 170, "Jocelin": 170}),
    "Mascotas": pd.Series([2, 9], ["Anel", "Jocelin"])
}

df = pd.DataFrame(Personas)
df
```

Out[22]:

	Peso	Altura	Mascotas
Anel	60	145	2.0
Chucho	74	170	NaN
Emilio	72	169	NaN
Jocelin	73	170	9.0

Es posible forzar el DataFrame a que presente determinadas columnas y en orden determinado.

In [24]:
```
# Creación de un DataFrame e inicializarlo con un diccionario de objeto de series.
Personas = {
    "Peso": pd.Series([72, 60, 74, 73], ["Emilio", "Anel", "Chucho", "Jocelin"]),
    "Altura": pd.Series({"Emilio": 169, "Anel": 145, "Chucho": 170, "Jocelin": 170}),
    "Mascotas": pd.Series([2, 9], ["Anel", "Jocelin"])
}

df = pd.DataFrame(
    Personas,
    columns = ["Altura", "Peso"],
    index = ["Chucho", "Emilio"])
df
```

Out[24]:

	Altura	Peso
Chucho	170	74

```
df = pd.DataFrame(
    Personas,
    columns = ["Altura", "Peso"],
    index = ["Chucho", "Emilio"])
df
```

Out[24]:

	Altura	Peso
Chucho	170	74
Emilio	169	72

In [29]:
```
# Creación de un DataFrame e inicializarlo con una lista de listas de Python.
# Nota: Deben especificarse las columnas e indices por separado.
Valores = [
    [169, 3, 72],
    [145, 2, 60],
    [170, 1, 74]
]

df = pd.DataFrame(
    Valores,
    columns = ["Altura", "Mascotas", "Peso"],
    index = ["Jocelin", "Emilio", "Anel"]
)
df
```

Out[29]:

	Altura	Mascotas	Peso
Jocelin	169	3	72
Emilio	145	2	60
Anel	170	1	74

In [ ]:
```
```