Ejercicio Pandas-01

Tienes un DataFrame con ventas de diferentes productos por meses.

Extrae la media de ventas para cada producto, solo para los meses donde la venta fue mayor que 50, y ordena el resultado de mayor a menor.

```
data = {
  'Producto': ['A', 'B', 'C', 'A', 'B', 'C', 'A', 'B', 'C'],
  'Mes': ['Enero', 'Enero', 'Enero', 'Febrero', 'Febrero', 'Febrero', 'Marzo', 'Marzo', 'Marzo'],
  'Ventas': [45, 60, 70, 55, 40, 80, 65, 70, 30]
}
```

Ejercicio Pandas-02

Crea una Series con índices personalizados a partir de un set, luego concaténala con un DataFrame por columnas, y finalmente, reemplaza todos los valores NaN por la media de la columna correspondiente.

```
serie = pd.Series(set([10, 20, 30]), index=['x', 'y', 'z'])
df = pd.DataFrame({'A':[1, 2, None], 'B':[4, None, 6]}, index=['x', 'y', 'w'])
```

Ejercicio Pandas-03

Dado un DataFrame, selecciona todas las filas cuyo índice sea par, y dentro de esas filas, filtra las columnas que tengan una media mayor que 3.

```
df = pd.DataFrame({
  'X': [1, 4, 7, 10],
  'Y': [2, 5, 8, 11],
  'Z': [3, 6, 9, 12]
}, index=[0, 1, 2, 3])
```

Ejercicio Pandas-04

Carga un DataFrame desde un CSV creado en memoria (usando pd.read_csv con StringIO), conviértelo a fechas, y crea una columna nueva con el mes extraído de la fecha. Luego crea una tabla dinámica con la media de una columna numérica por mes.

```
from io import StringIO
```

```
csv_data = """id,fecha,valor
1,2023-01-10,100
2,2023-02-15,200
3,2023-01-20,150
4,2023-03-05,300
""".
```

Ejercicio Pandas-05

Dado un DataFrame con categorías y valores, renombra las columnas quitando espacios y conviértelas a minúsculas, convierte la columna categoría a tipo category, y muestra la frecuencia de cada categoría.

```
df = pd.DataFrame({
  ' Cat ': ['a', 'b', 'a', 'c', 'b', 'a'],
  'Valor': [10, 20, 10, 30, 20, 10]
}).
```

Ejercicio Pandas-06

Tienes dos DataFrames con nombres y edades, pero con diferentes columnas para nombre. Realiza un merge con outer join y rellena los valores NaN con 'Desconocido' para nombres y con la media para edades

```
df1 = pd.DataFrame({'nombre1': ['Ana', 'Luis'], 'edad1': [23, 45]})
df2 = pd.DataFrame({'nombre2': ['Ana', 'Juan'], 'edad2': [23, 30]}).
```

Ejercicio Pandas-07

Crea un DataFrame con índices de fecha y varias columnas numéricas. Usa .apply() para normalizar cada columna (restar media y dividir por desviación estándar) y filtra las filas donde alguna columna esté fuera del rango [-1, 1].

```
fechas = pd.date_range('2023-01-01', periods=5)
df = pd.DataFrame(np.random.randint(1, 100, (5,3)), columns=list('ABC'), index=fechas).
```

Ejercicio Pandas-08

Dado un DataFrame con columnas de strings, convierte todas a mayúsculas, elimina las filas que contengan el string 'ERROR', y ordena por la columna 'ID' descendente.

```
df = pd.DataFrame({
  'ID': [3, 2, 1],
  'Status': ['ok', 'error', 'ok'],
  'Name': ['Ana', 'Luis', 'Juan']
}).
```

Ejercicio Pandas-09

Genera un DataFrame de 6 filas con columnas 'Grupo' y 'Puntaje', agrupa por 'Grupo' y calcula la suma, media y desviación estándar, luego convierte el resultado a un DataFrame plano (sin índices jerárquicos)

```
df = pd.DataFrame({
  'Grupo': ['X', 'Y', 'X', 'Y', 'X', 'Z'],
  'Puntaje': [10, 20, 15, 25, 10, 5]
}).
```

Ejercicio Pandas-10

Crea un DataFrame con columnas desordenadas y con espacios en los nombres. Reordénalas alfabéticamente, elimina la columna menos importante, y modifica un valor específico usando .iloc

```
df = pd.DataFrame({
  ' C ': [1, 2],
  'A': [3, 4],
  ' B': [5, 6]
}).
```

Ejercicio Pandas-11

A partir de un DataFrame, crea una nueva columna con valores categóricos en mayúsculas, reemplaza los valores NaN por un valor definido, y cuenta cuántos valores únicos tiene esa columna

```
df = pd.DataFrame({
  'Categoria': ['a', None, 'b', 'a', 'c', None]
}).
```

Ejercicio Pandas-12

Crea un DataFrame a partir de una lista de diccionarios, luego ordena primero por dos columnas (una ascendente y otra descendente), y extrae las últimas 3 filas

```
lista = [
  {'Nombre':'Ana', 'Edad':25, 'Puntaje':90},
  {'Nombre':'Luis', 'Edad':30, 'Puntaje':80},
  {'Nombre':'Juan', 'Edad':25, 'Puntaje':95},
  {'Nombre':'Maria', 'Edad':22, 'Puntaje':88},
  {'Nombre':'Carmen', 'Edad':30, 'Puntaje':85},
].
```

Ejercicio Pandas-13

Tienes dos DataFrames con columnas parcialmente coincidentes, realiza un merge con inner join usando columnas con nombres distintos, y luego elimina filas duplicadas.

```
df1 = pd.DataFrame({'ID1': [1,2,3], 'Valor': [100, 200, 300]})
df2 = pd.DataFrame({'ID2': [2,3,4], 'Valor2': [250, 350, 450]})
```

Ejercicio Pandas-14

Crea un DataFrame con datos numéricos, elimina las filas que contengan NaN, calcula el coeficiente de correlación entre todas las columnas, y finalmente, imprime el valor más alto de correlación diferente de 1.

```
df = pd.DataFrame({
  'A':[1, 2, 3, 4, np.nan],
  'B':[2, 3, 4, 5, 6],
  'C':[5, 6, 7, np.nan, 9]
})
```

Ejercicio Pandas-15

Crea un DataFrame con columnas de texto y numéricas. Aplica una función lambda que convierte a mayúsculas solo las columnas de texto y que a las columnas numéricas les suma 100.

```
df = pd.DataFrame({
  'Nombre': ['Ana', 'Luis', 'Juan'],
  'Edad': [25, 30, 22],
  'Ciudad': ['madrid', 'barcelona', 'sevilla']
})
```

Ejercicio Pandas-16

Crea un DataFrame, cambia el índice por una columna, concatena horizontalmente otro DataFrame, y después elimina una fila por índice usando .drop()

```
df1 = pd.DataFrame({
  'ID': [1, 2, 3],
  'Valor': [10, 20, 30]
})
df2 = pd.DataFrame({
  'Extra': ['a', 'b', 'c']
}).
```

Ejercicio Pandas-17

Lee un CSV desde un string con separador ; y sin cabecera, asigna nombres a las columnas, cambia el tipo de una columna a category y filtra por valores que están en una lista

```
from io import StringIO

csv_data = """Ana;25;Madrid
Luis;30;Barcelona
Juan;22;Sevilla
Maria;28;Madrid
""".
```

Ejercicio Pandas-18

Genera un DataFrame, ordena por columnas de forma descendente, crea un nuevo DataFrame con la media y suma agrupada por una columna, y escribe el resultado a un archivo Excel (en memoria con BytesIO).

```
import io

df = pd.DataFrame({

  'Categoria': ['A', 'B', 'A', 'B', 'C'],

  'Valor': [10, 20, 15, 25, 30]

}).
```

Ejercicio Pandas-19

Dado un DataFrame con fechas en formato string, convierte la columna a datetime, extrae el año, filtra los datos para un año específico y calcula el conteo de registros para cada mes.

```
df = pd.DataFrame({

  'fecha': ['2020-01-05', '2020-02-15', '2021-01-10', '2020-03-22', '2021-04-30'],

  'valor': [10, 20, 30, 40, 50]

}).
```

Ejercicio Pandas-20

Crea un DataFrame con columnas numéricas y una categórica, usa pivot_table para obtener la suma y el promedio agrupando por la columna categórica, y añade totales con margins=True

```
df = pd.DataFrame({

  'Categoria': ['X', 'Y', 'X', 'Y', 'Z'],

  'Valor1': [10, 20, 30, 40, 50],

  'Valor2': [1, 2, 3, 4, 5]

}).
```