

Graficar Calificaciones Excel



Tengo un archivo llamado "calificaciones_alumnos.xlsx", el cual tiene un campo llamado "Nombre" y varias columnas de calificaciones llamadas Mat_CalculoIntegral, Mat_ProgramacionOOP y Mat_EstructuraDatos. Ayúdame a crear un programa en Python para graficar las calificaciones para cada alumno pero no quiero que se encimen sus etiquetas en el eje X, al inicio del código agrega un comentario explicando lo que hace el programa y documenta las líneas importantes del código



Aquí tienes un programa en Python que carga el archivo "calificaciones_alumnos.xlsx", lee los datos, y grafica las calificaciones de cada alumno asegurándose de que las etiquetas en el eje X no se encimen. He agregado comentarios explicativos al inicio y en las líneas importantes del código.

python

```
# Este programa carga un archivo de Excel llamado "calificaciones_alumnos.xlsx",
# lee los datos de calificaciones de los alumnos y crea un gráfico de barras para
# visualizar las calificaciones en diferentes materias sin que las etiquetas en el
# eje X se encimen.

import pandas as pd
import matplotlib.pyplot as plt

# Cargar el archivo Excel
file_path = 'calificaciones_alumnos.xlsx'
df = pd.read_excel(file_path)

# Nombres de los alumnos
nombres = df['Nombre']

# Calificaciones en diferentes materias
calculo_integral = df['Mat_CalculoIntegral']
programacion_oop = df['Mat_ProgramacionOOP']
estructura_datos = df['Mat_EstructuraDatos']

# Configurar el gráfico de barras
fig, ax = plt.subplots(figsize=(10, 6))

# Posiciones de las barras en el eje X
bar_width = 0.2
bar_positions = list(range(len(nombres)))

# Dibujar las barras para cada materia
ax.bar(bar_positions, calculo_integral, width=bar_width, label='Cálculo Integral',
align='center')
ax.bar([p + bar_width for p in bar_positions], programacion_oop, width=bar_width,
label='Programación OOP', align='center')
ax.bar([p + 2 * bar_width for p in bar_positions], estructura_datos, width=bar_width,
label='Estructura de Datos', align='center')
```

```
# Configuración de las etiquetas en el eje X
ax.set_xticks([p + bar_width for p in bar_positions])
ax.set_xticklabels(nombres, rotation=45, ha='right')

# Añadir leyenda y etiquetas
ax.set_xlabel('Alumnos')
ax.set_ylabel('Calificaciones')
ax.set_title('Calificaciones de los Alumnos en Diferentes Materias')
ax.legend()

# Mostrar gráfico
plt.tight_layout()
plt.show()
```

Explicación del Código:

1. Importar bibliotecas:

- `pandas` para manejar los datos de Excel.
- `matplotlib.pyplot` para graficar los datos.

2. Cargar el archivo Excel:

python

```
file_path = 'calificaciones_alumnos.xlsx'
df = pd.read_excel(file_path)
```

Esta sección carga el archivo Excel y lo convierte en un DataFrame de pandas.

3. Extraer datos:

python

```
nombres = df['Nombre']
calculo_integral = df['Mat_CalculoIntegral']
programacion_oop = df['Mat_ProgramacionOOP']
estructura_datos = df['Mat_EstructuraDatos']
```

Extrae las columnas relevantes del DataFrame.

4. Configurar el gráfico de barras:

python

```
fig, ax = plt.subplots(figsize=(10, 6))
bar_width = 0.2
bar_positions = list(range(len(nombres)))
```

Configura la figura y las posiciones de las barras.

5. Dibujar las barras:

python

```
ax.bar(bar_positions, calculo_integral, width=bar_width, label='Cálculo Integral',
align='center')
ax.bar([p + bar_width for p in bar_positions], programacion_oop, width=bar_width,
label='Programación OOP', align='center')
ax.bar([p + 2 * bar_width for p in bar_positions], estructura_datos, width=bar_width,
label='Estructura de Datos', align='center')
```

Dibuja las barras para cada materia, desplazándolas para que no se encimen.

6. Configurar etiquetas en el eje X:

python

```
ax.set_xticks([p + bar_width for p in bar_positions])
ax.set_xticklabels(nombres, rotation=45, ha='right')
```

Ajusta las etiquetas para que no se encimen y sean legibles.

7. Añadir leyenda y etiquetas:

python

```
ax.set_xlabel('Alumnos')
ax.set_ylabel('Calificaciones')
ax.set_title('Calificaciones de los Alumnos en Diferentes Materias')
ax.legend()
```

Añade etiquetas y leyenda al gráfico para mayor claridad.

8. Mostrar el gráfico:

python

```
plt.tight_layout()
plt.show()
```

Ajusta el diseño y muestra el gráfico.



Como importo las bibliotecas del programa



Para importar las bibliotecas necesarias para el programa, asegúrate de tener `pandas` y `matplotlib` instalados en tu entorno de Python. Puedes instalarlas usando `pip` si aún no lo has hecho:

```
sh
```

```
pip install pandas matplotlib
```

Luego, en el código de Python, asegúrate de incluir las importaciones al inicio del script:

```
python
```

```
import pandas as pd
import matplotlib.pyplot as plt
```

Aquí está el código completo con las importaciones incluidas:

```
python
```

```
# Este programa carga un archivo de Excel llamado "calificaciones_alumnos.xlsx",
# lee los datos de calificaciones de los alumnos y crea un gráfico de barras para
# visualizar las calificaciones en diferentes materias sin que las etiquetas en el
# eje X se encimen.

import pandas as pd
import matplotlib.pyplot as plt

# Cargar el archivo Excel
file_path = 'calificaciones_alumnos.xlsx'
df = pd.read_excel(file_path)

# Nombres de los alumnos
nombres = df['Nombre']

# Calificaciones en diferentes materias
calculo_integral = df['Mat_CalculoIntegral']
programacion_oop = df['Mat_ProgramacionOOP']
estructura_datos = df['Mat_EstructuraDatos']

# Configurar el gráfico de barras
fig, ax = plt.subplots(figsize=(10, 6))

# Posiciones de las barras en el eje X
bar_width = 0.2
bar_positions = list(range(len(nombres)))

# Dibujar las barras para cada materia
ax.bar(bar_positions, calculo_integral, width=bar_width, label='Cálculo Integral',
align='center')
ax.bar([p + bar_width for p in bar_positions], programacion_oop, width=bar_width,
label='Programación OOP', align='center')
ax.bar([p + 2 * bar_width for p in bar_positions], estructura_datos, width=bar_width,
label='Estructura de Datos', align='center')

# Configuración de las etiquetas en el eje X
ax.set_xticks([p + bar_width for p in bar_positions])
ax.set_xticklabels(nombres, rotation=45, ha='right')
```

```
# Añadir leyenda y etiquetas
ax.set_xlabel('Alumnos')
ax.set_ylabel('Calificaciones')
ax.set_title('Calificaciones de los Alumnos en Diferentes Materias')
ax.legend()

# Mostrar gráfico
plt.tight_layout()
plt.show()
```

Con esto, el programa importará las bibliotecas necesarias y estará listo para ejecutarse correctamente.



Tengo un archivo llamado "calificaciones_alumnos.xlsx", el cual tiene un campo llamado "Nombre" y varias columnas de calificaciones llamadas Mat_CalculoIntegral, Mat_ProgramacionOOP y Mat_EstructuraDatos. Ayúdame a crear un programa en Python para agregar una columna llamada Mat_Fisica que contenga valores aleatorios entre 0 y 100 con un decimal, al inicio del código agrega un comentario explicando lo que hace el programa y documenta las líneas importantes del código



Aquí tienes un programa en Python que carga el archivo "calificaciones_alumnos.xlsx", agrega una nueva columna llamada "Mat_Fisica" con valores aleatorios entre 0 y 100 con un decimal, y guarda el archivo actualizado. He agregado comentarios explicativos al inicio y en las líneas importantes del código.

python

```
# Este programa carga un archivo de Excel llamado "calificaciones_alumnos.xlsx",
# agrega una nueva columna llamada "Mat_Fisica" con valores aleatorios entre 0 y 100
# con un decimal, y guarda el archivo actualizado.

import pandas as pd
import numpy as np

# Cargar el archivo Excel
file_path = 'calificaciones_alumnos.xlsx'
df = pd.read_excel(file_path)

# Generar valores aleatorios para la columna "Mat_Fisica"
np.random.seed(0) # Fijar la semilla para reproducibilidad
df['Mat_Fisica'] = np.round(np.random.uniform(0, 100, len(df)), 1)

# Guardar el archivo actualizado
df.to_excel('calificaciones_alumnos_actualizado.xlsx', index=False)
```

Explicación del Código:

1. Importar bibliotecas:

python

```
import pandas as pd
import numpy as np
```

Importa las bibliotecas necesarias: `pandas` para manejar los datos de Excel y `numpy` para generar valores aleatorios.

2. Cargar el archivo Excel:

python

```
file_path = 'calificaciones_alumnos.xlsx'
df = pd.read_excel(file_path)
```

Carga el archivo Excel y lo convierte en un DataFrame de pandas.

3. Generar valores aleatorios para "Mat_Fisica":

python

```
np.random.seed(0) # Fijar la semilla para reproducibilidad
df['Mat_Fisica'] = np.round(np.random.uniform(0, 100, len(df)), 1)
```

- `np.random.seed(0)`: Fija la semilla para que los valores aleatorios sean reproducibles.
- `np.random.uniform(0, 100, len(df))`: Genera valores aleatorios entre 0 y 100 para cada fila del DataFrame.
- `np.round(..., 1)`: Redondea los valores a un decimal.
- `df['Mat_Fisica'] = ...`: Añade los valores generados como una nueva columna llamada "Mat_Fisica".

4. Guardar el archivo actualizado:

python

```
df.to_excel('calificaciones_alumnos_actualizado.xlsx', index=False)
```

Guarda el DataFrame actualizado en un nuevo archivo Excel llamado "calificaciones_alumnos_actualizado.xlsx".

Este programa añadirá la columna "Mat_Fisica" con valores aleatorios a tu archivo de calificaciones y guardará los cambios en un nuevo archivo Excel.



Para el nuevo archivo generado quiero que me ordene la tabla por medio del Nombre, al inicio del código agrega un comentario explicando lo que hace el programa y documenta las líneas importantes del código



Aquí tienes el programa en Python que carga el archivo "calificaciones_alumnos.xlsx", agrega una nueva columna llamada "Mat_Fisica" con valores aleatorios entre 0 y 100 con un decimal, ordena la tabla por el campo "Nombre" y guarda el archivo actualizado. He agregado comentarios explicativos al inicio y en las líneas importantes del código.

python

```
# Este programa carga un archivo de Excel llamado "calificaciones_alumnos.xlsx",
# agrega una nueva columna llamada "Mat_Fisica" con valores aleatorios entre 0 y 100
# con un decimal, ordena la tabla por el campo "Nombre" y guarda el archivo actualizado.

import pandas as pd
import numpy as np

# Cargar el archivo Excel
file_path = 'calificaciones_alumnos.xlsx'
df = pd.read_excel(file_path)

# Generar valores aleatorios para la columna "Mat_Fisica"
np.random.seed(0) # Fijar la semilla para reproducibilidad
df['Mat_Fisica'] = np.round(np.random.uniform(0, 100, len(df)), 1)

# Ordenar el DataFrame por el campo "Nombre"
df_sorted = df.sort_values(by='Nombre')

# Guardar el archivo actualizado
df_sorted.to_excel('calificaciones_alumnos_actualizado.xlsx', index=False)
```

Explicación del Código:

1. Importar bibliotecas:

python

```
import pandas as pd
import numpy as np
```

Importa las bibliotecas necesarias: `pandas` para manejar los datos de Excel y `numpy` para generar valores aleatorios.

2. Cargar el archivo Excel:

python

```
file_path = 'calificaciones_alumnos.xlsx'
df = pd.read_excel(file_path)
```

Carga el archivo Excel y lo convierte en un DataFrame de pandas.

3. Generar valores aleatorios para "Mat_Fisica":

python

```
np.random.seed(0) # Fijar la semilla para reproducibilidad
df['Mat_Fisica'] = np.round(np.random.uniform(0, 100, len(df)), 1)
```

- `np.random.seed(0)`: Fija la semilla para que los valores aleatorios sean reproducibles.
- `np.random.uniform(0, 100, len(df))`: Genera valores aleatorios entre 0 y 100 para cada fila del DataFrame.
- `np.round(..., 1)`: Redondea los valores a un decimal.
- `df['Mat_Fisica'] = ...`: Añade los valores generados como una nueva columna llamada "Mat_Fisica".

4. Ordenar el DataFrame por el campo "Nombre":

python

```
df_sorted = df.sort_values(by='Nombre')
```

Ordena el DataFrame por la columna "Nombre".

5. Guardar el archivo actualizado:

python

```
df_sorted.to_excel('calificaciones_alumnos_actualizado.xlsx', index=False)
```

Guarda el DataFrame ordenado en un nuevo archivo Excel llamado "calificaciones_alumnos_actualizado.xlsx".

Este programa añadirá la columna "Mat_Fisica" con valores aleatorios a tu archivo de calificaciones, ordenará los datos por el nombre de los alumnos y guardará los cambios en un nuevo archivo Excel.



Para el nuevo archivo generado quiero que me ordene la tabla por medio del Nombre, al inicio del código agrega un comentario explicando lo que hace el programa y documenta las líneas

importantes del código



Aquí tienes el programa en Python que carga el archivo "calificaciones_alumnos.xlsx", agrega una nueva columna llamada "Mat_Fisica" con valores aleatorios entre 0 y 100 con un decimal, ordena la tabla por el campo "Nombre", y guarda el archivo actualizado. He agregado comentarios explicativos al inicio y en las líneas importantes del código.

python

```
# Este programa carga un archivo de Excel llamado "calificaciones_alumnos.xlsx",
# agrega una nueva columna llamada "Mat_Fisica" con valores aleatorios entre 0 y 100
# con un decimal, ordena la tabla por el campo "Nombre", y guarda el archivo actualizado.

import pandas as pd
import numpy as np

# Cargar el archivo Excel
file_path = 'calificaciones_alumnos.xlsx'
df = pd.read_excel(file_path)

# Generar valores aleatorios para la columna "Mat_Fisica"
np.random.seed(0) # Fijar la semilla para reproducibilidad
df['Mat_Fisica'] = np.round(np.random.uniform(0, 100, len(df)), 1)

# Ordenar el DataFrame por el campo "Nombre"
df_sorted = df.sort_values(by='Nombre')

# Guardar el archivo actualizado
df_sorted.to_excel('calificaciones_alumnos_actualizado.xlsx', index=False)
```

Explicación del Código:

1. Importar bibliotecas:

python

```
import pandas as pd
import numpy as np
```

Importa las bibliotecas necesarias: `pandas` para manejar los datos de Excel y `numpy` para generar valores aleatorios.

2. Cargar el archivo Excel:

python

```
file_path = 'calificaciones_alumnos.xlsx'
```

```
df = pd.read_excel(file_path)
```

Carga el archivo Excel y lo convierte en un DataFrame de pandas.

3. Generar valores aleatorios para "Mat_Fisica":

python

```
np.random.seed(0) # Fijar la semilla para reproducibilidad
df['Mat_Fisica'] = np.round(np.random.uniform(0, 100, len(df)), 1)
```

- `np.random.seed(0)`: Fija la semilla para que los valores aleatorios sean reproducibles.
- `np.random.uniform(0, 100, len(df))`: Genera valores aleatorios entre 0 y 100 para cada fila del DataFrame.
- `np.round(..., 1)`: Redondea los valores a un decimal.
- `df['Mat_Fisica'] = ...`: Añade los valores generados como una nueva columna llamada "Mat_Fisica".

4. Ordenar el DataFrame por el campo "Nombre":

python

```
df_sorted = df.sort_values(by='Nombre')
```

Ordena el DataFrame por la columna "Nombre".

5. Guardar el archivo actualizado:

python

```
df_sorted.to_excel('calificaciones_alumnos_actualizado.xlsx', index=False)
```

Guarda el DataFrame ordenado en un nuevo archivo Excel llamado "calificaciones_alumnos_actualizado.xlsx".

Este programa añadirá la columna "Mat_Fisica" con valores aleatorios a tu archivo de calificaciones, ordenará los datos por el nombre de los alumnos y guardará los cambios en un nuevo archivo Excel.



Quiero saber cuántos registros y cuántos campos tiene mi tabla, al inicio del código agrega un comentario explicando lo que hace el programa y documenta las líneas importantes del código



Aquí tienes el programa en Python que carga el archivo "calificaciones_alumnos.xlsx", agrega una nueva columna llamada "Mat_Fisica" con valores aleatorios entre 0 y 100 con un decimal, ordena la tabla por el campo "Nombre", imprime la cantidad de registros y campos en la tabla, y guarda el archivo actualizado. He agregado comentarios explicativos al inicio y en las líneas importantes del código.

python

```
# Este programa carga un archivo de Excel llamado "calificaciones_alumnos.xlsx",
# agrega una nueva columna llamada "Mat_Fisica" con valores aleatorios entre 0 y 100
# con un decimal, ordena la tabla por el campo "Nombre", imprime la cantidad de registros
# y campos en la tabla, y guarda el archivo actualizado.

import pandas as pd
import numpy as np

# Cargar el archivo Excel
file_path = 'calificaciones_alumnos.xlsx'
df = pd.read_excel(file_path)

# Generar valores aleatorios para la columna "Mat_Fisica"
np.random.seed(0) # Fijar la semilla para reproducibilidad
df['Mat_Fisica'] = np.round(np.random.uniform(0, 100, len(df)), 1)

# Ordenar el DataFrame por el campo "Nombre"
df_sorted = df.sort_values(by='Nombre')

# Imprimir la cantidad de registros y campos
num_registros = df_sorted.shape[0]
num_campos = df_sorted.shape[1]
print(f'Cantidad de registros: {num_registros}')
print(f'Cantidad de campos: {num_campos}')

# Guardar el archivo actualizado
df_sorted.to_excel('calificaciones_alumnos_actualizado.xlsx', index=False)
```

Explicación del Código:

1. Importar bibliotecas:

python

```
import pandas as pd
import numpy as np
```

Importa las bibliotecas necesarias: `pandas` para manejar los datos de Excel y `numpy` para generar valores aleatorios.

2. Cargar el archivo Excel:

python

```
file_path = 'calificaciones_alumnos.xlsx'
df = pd.read_excel(file_path)
```

Carga el archivo Excel y lo convierte en un DataFrame de pandas.

3. Generar valores aleatorios para "Mat_Fisica":

python

```
np.random.seed(0) # Fijar la semilla para reproducibilidad
df['Mat_Fisica'] = np.round(np.random.uniform(0, 100, len(df)), 1)
```

- `np.random.seed(0)`: Fija la semilla para que los valores aleatorios sean reproducibles.
- `np.random.uniform(0, 100, len(df))`: Genera valores aleatorios entre 0 y 100 para cada fila



Quiero saber cuántos registros y cuántos campos tiene mi tabla, al inicio del código agrega un comentario explicando lo que hace el programa y documenta las líneas importantes del código



Aquí tienes el programa en Python que carga el archivo "calificaciones_alumnos.xlsx", agrega una nueva columna llamada "Mat_Fisica" con valores aleatorios entre 0 y 100 con un decimal, ordena la tabla por el campo "Nombre", imprime la cantidad de registros y campos en la tabla, y guarda el archivo actualizado. He agregado comentarios explicativos al inicio y en las líneas importantes del código.

python

```
# Este programa carga un archivo de Excel llamado "calificaciones_alumnos.xlsx",
# agrega una nueva columna llamada "Mat_Fisica" con
```



Quiero saber cuántos registros y cuántos campos tiene mi tabla, al inicio del código agrega un comentario explicando lo que hace el programa y documenta las líneas importantes del código



Aquí tienes el programa en Python que carga el archivo "calificaciones_alumnos.xlsx", agrega una nueva columna llamada "Mat_Fisica" con valores aleatorios entre 0 y 100 con un decimal, ordena la tabla por el campo "Nombre", imprime la cantidad de registros y campos en la tabla, y guarda el archivo actualizado. He agregado comentarios explicativos al inicio y en las líneas importantes del código.

python

```
# Este programa carga un archivo de Excel llamado "calificaciones_alumnos.xlsx",
# agrega una nueva columna llamada "Mat_Fisica" con valores aleatorios entre 0 y 100
# con un decimal, ordena la tabla por el campo "Nombre", imprime la cantidad de registros
# y campos en la tabla, y guarda el archivo actualizado.

import pandas as pd
import numpy as np

# Cargar el archivo Excel
file_path = 'calificaciones_alumnos.xlsx'
df = pd.read_excel(file_path)

# Generar valores aleatorios para la columna "Mat_Fisica"
np.random.seed(0) # Fijar la semilla para reproducibilidad
df['Mat_Fisica'] = np.round(np.random.uniform(0, 100, len(df)), 1)

# Ordenar el DataFrame por el campo "Nombre"
df_sorted = df.sort_values(by='Nombre')

# Imprimir la cantidad de registros y campos
num_registros = df_sorted.shape[0]
num_campos = df_sorted.shape[1]
print(f'Cantidad de registros: {num_registros}')
print(f'Cantidad de campos: {num_campos}')

# Guardar el archivo actualizado
df_sorted.to_excel('calificaciones_alumnos_actualizado.xlsx', index=False)
```

Explicación del Código:

1. Importar bibliotecas:

python

```
import pandas as pd
import numpy as np
```

Importa las bibliotecas necesarias: `pandas` para manejar los datos de Excel y `numpy` para generar valores aleatorios.

2. Cargar el archivo Excel:

python

```
file_path = 'calificaciones_alumnos.xlsx'
df = pd.read_excel(file_path)
```

Carga el archivo Excel y lo convierte en un DataFrame de pandas.

3. Generar valores aleatorios para "Mat_Fisica":

python

```
np.random.seed(0) # Fijar la semilla para reproducibilidad
df['Mat_Fisica'] = np.round(np.random.uniform(0, 100, len(df)), 1)
```

- `np.random.seed(0)`: Fija la semilla para que los valores aleatorios sean reproducibles.
- `np.random.uniform(0, 100, len(df))`: Genera valores aleatorios entre 0 y 100 para cada fila del DataFrame.
- `np.round(..., 1)`: Redondea los valores a un decimal.
- `df['Mat_Fisica'] = ...`: Añade los valores generados como una nueva columna llamada "Mat_Fisica".

4. Ordenar el DataFrame por el campo "Nombre":

python

```
df_sorted = df.sort_values(by='Nombre')
```

Ordena el DataFrame por la columna "Nombre".

5. Imprimir la cantidad de registros y campos:

python

```
num_registros = df_sorted.shape[0]
num_campos = df_sorted.shape[1]
print(f'Cantidad de registros: {num_registros}')
print(f'Cantidad de campos: {num_campos}')
```

- `num_registros = df_sorted.shape[0]`: Obtiene la cantidad de registros (filas) en el DataFrame.
- `num_campos = df_sorted.shape[1]`: Obtiene la cantidad de campos (columnas) en el DataFrame.
- `print(...)`: Imprime la cantidad de registros y campos.

6. Guardar el archivo actualizado:

python

```
df_sorted.to_excel('calificaciones_alumnos_actualizado.xlsx', index=False)
```

Guarda el DataFrame ordenado en un nuevo archivo Excel llamado "calificaciones_alumnos_actualizado.xlsx".

Este programa añadirá la columna "Mat_Fisica" con valores aleatorios a tu archivo de calificaciones, ordenará los datos por el nombre de los alumnos, imprimirá la cantidad de registros y campos, y guardará los cambios en un nuevo archivo Excel.



Quiero saber qué campos son numéricos, al inicio del código agrega un comentario explicando lo que hace el programa y documenta las líneas importantes del código



Aquí tienes el programa en Python que carga el archivo "calificaciones_alumnos.xlsx", agrega una nueva columna llamada "Mat_Fisica" con valores aleatorios entre 0 y 100 con un decimal, ordena la tabla por el campo "Nombre", imprime la cantidad de registros y campos en la tabla, imprime los nombres de los campos numéricos, y guarda el archivo actualizado. He agregado comentarios explicativos al inicio y en las líneas importantes del código.

python

```
# Este programa carga un archivo de Excel llamado "calificaciones_alumnos.xlsx",
# agrega una nueva columna llamada "Mat_Fisica" con valores aleatorios entre 0 y 100
# con un decimal, ordena la tabla por el campo "Nombre", imprime la cantidad de registros
# y campos en la tabla, identifica los campos numéricos, e imprime sus nombres, y guarda
# el archivo actualizado.

import pandas as pd
import numpy as np

# Cargar el archivo Excel
file_path = 'calificaciones_alumnos.xlsx'
df = pd.read_excel(file_path)

# Generar valores aleatorios para la columna "Mat_Fisica"
np.random.seed(0) # Fijar la semilla para reproducibilidad
df['Mat_Fisica'] = np.round(np.random.uniform(0, 100, len(df)), 1)

# Ordenar el DataFrame por el campo "Nombre"
df_sorted = df.sort_values(by='Nombre')

# Imprimir la cantidad de registros y campos
num_registros = df_sorted.shape[0]
num_campos = df_sorted.shape[1]
print(f'Cantidad de registros: {num_registros}')
print(f'Cantidad de campos: {num_campos}')
```

```
# Identificar e imprimir los campos numéricos
numeric_fields = df_sorted.select_dtypes(include=[np.number]).columns.tolist()
print(f'Campos numéricos: {numeric_fields}')

# Guardar el archivo actualizado
df_sorted.to_excel('calificaciones_alumnos_actualizado.xlsx', index=False)
```

Explicación del Código:

1. Importar bibliotecas:

python

```
import pandas as pd
import numpy as np
```

Importa las bibliotecas necesarias: `pandas` para manejar los datos de Excel y `numpy` para generar valores aleatorios.

2. Cargar el archivo Excel:

python

```
file_path = 'calificaciones_alumnos.xlsx'
df = pd.read_excel(file_path)
```

Carga el archivo Excel y lo convierte en un DataFrame de pandas.

3. Generar valores aleatorios para "Mat_Fisica":

python

```
np.random.seed(0) # Fijar la semilla para reproducibilidad
df['Mat_Fisica'] = np.round(np.random.uniform(0, 100, len(df)), 1)
```

- `np.random.seed(0)`: Fija la semilla para que los valores aleatorios sean reproducibles.
- `np.random.uniform(0, 100, len(df))`: Genera valores aleatorios entre 0 y 100 para cada fila del DataFrame.
- `np.round(..., 1)`: Redondea los valores a un decimal.
- `df['Mat_Fisica'] = ...`: Añade los valores generados como una nueva columna llamada "Mat_Fisica".

4. Ordenar el DataFrame por el campo "Nombre":

python

```
df_sorted = df.sort_values(by='Nombre')
```

Ordena el DataFrame por la columna "Nombre".

5. Imprimir la cantidad de registros y campos:

python

```
num_registros = df_sorted.shape[0]
num_campos = df_sorted.shape[1]
print(f'Cantidad de registros: {num_registros}')
print(f'Cantidad de campos: {num_campos}')
```

- `num_registros = df_sorted.shape[0]`: Obtiene la cantidad de registros (filas) en el DataFrame.
- `num_campos = df_sorted.shape[1]`: Obtiene la cantidad de campos (columnas) en el DataFrame.
- `print(...)`: Imprime la cantidad de registros y campos.

6. Identificar e imprimir los campos numéricos:

python

```
numeric_fields = df_sorted.select_dtypes(include=[np.number]).columns.tolist()
print(f'Campos numéricos: {numeric_fields}')
```

- `df_sorted.select_dtypes(include=[np.number])`: Selecciona las columnas del DataFrame que son de tipo numérico.
- `.columns.tolist()`: Convierte los nombres de las columnas seleccionadas en una lista.
- `print(...)`: Imprime los nombres de los campos numéricos.

7. Guardar el archivo actualizado:

python

```
df_sorted.to_excel('calificaciones_alumnos_actualizado.xlsx', index=False)
```

Guarda el DataFrame ordenado en un nuevo archivo Excel llamado "calificaciones_alumnos_actualizado.xlsx".

Este programa añadirá la columna "Mat_Fisica" con valores aleatorios a tu archivo de calificaciones, ordenará los datos por el nombre de los alumnos, imprimirá la cantidad de registros y campos, identificará e imprimirá los campos numéricos, y guardará los cambios en un nuevo archivo Excel.



Genera 15 prompts detallados para automatizar procesos y obtener respuestas significativas al documento de Excel. Para cada prompt crea un archivo .py con la documentación interna correspondiente



Voy a generar 15 prompts detallados para automatizar procesos y obtener respuestas significativas del documento de Excel "calificaciones_alumnos.xlsx". Para cada prompt, crearé un archivo `.py` con la documentación interna correspondiente.

1. Leer y Mostrar el Archivo de Excel

Prompt: Crear un programa en Python que lea y muestre el contenido del archivo "calificaciones_alumnos.xlsx".

Archivo: `read_excel.py`

python

```
# Este programa carga y muestra el contenido de un archivo de Excel llamado
"calificaciones_alumnos.xlsx".

import pandas as pd

# Cargar el archivo Excel
file_path = 'calificaciones_alumnos.xlsx'
df = pd.read_excel(file_path)

# Mostrar el contenido del archivo
print(df)
```

2. Contar Registros y Campos

Prompt: Crear un programa en Python que cuente y muestre la cantidad de registros y campos en el archivo "calificaciones_alumnos.xlsx".

Archivo: `count_records_fields.py`

python

```
# Este programa cuenta y muestra la cantidad de registros y campos en un archivo de Excel
llamado "calificaciones_alumnos.xlsx".

import pandas as pd

# Cargar el archivo Excel
file_path = 'calificaciones_alumnos.xlsx'
df = pd.read_excel(file_path)

# Contar registros y campos
num_registros = df.shape[0]
num_campos = df.shape[1]
```

```
# Mostrar la cantidad de registros y campos
print(f'Cantidad de registros: {num_registros}')
print(f'Cantidad de campos: {num_campos}')
```

3. Identificar Campos Numéricos

Prompt: Crear un programa en Python que identifique y muestre los nombres de los campos numéricos en el archivo "calificaciones_alumnos.xlsx".

Archivo: `identify_numeric_fields.py`

python

```
# Este programa identifica y muestra los nombres de los campos numéricos en un archivo de
Excel llamado "calificaciones_alumnos.xlsx".

import pandas as pd
import numpy as np

# Cargar el archivo Excel
file_path = 'calificaciones_alumnos.xlsx'
df = pd.read_excel(file_path)

# Identificar campos numéricos
numeric_fields = df.select_dtypes(include=[np.number]).columns.tolist()

# Mostrar los campos numéricos
print(f'Campos numéricos: {numeric_fields}')
```

4. Ordenar por Nombre

Prompt: Crear un programa en Python que ordene el archivo "calificaciones_alumnos.xlsx" por el campo "Nombre" y guarde el archivo ordenado.

Archivo: `sort_by_name.py`

python

```
# Este programa ordena un archivo de Excel llamado "calificaciones_alumnos.xlsx" por el
campo "Nombre" y guarda el archivo ordenado.

import pandas as pd

# Cargar el archivo Excel
file_path = 'calificaciones_alumnos.xlsx'
df = pd.read_excel(file_path)

# Ordenar por el campo "Nombre"
df_sorted = df.sort_values(by='Nombre')

# Guardar el archivo ordenado
df_sorted.to_excel('calificaciones_alumnos_ordenado.xlsx', index=False)
```

5. Agregar Col

ChatGPT puede cometer errores. Considera verificar la información importante.