

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 los datos de un archivo Excel llamado "calificaciones_alumnos.xlsx" y grafica las calificaciones de cada alumno, asegurándose de que las etiquetas en el eje X no se encimen. El programa utiliza las bibliotecas `pandas` para la manipulación de datos y `matplotlib` para la creación de gráficos.

python

```
# Este programa carga un archivo Excel con calificaciones de alumnos y grafica dichas calificaciones.
# Se asegura de que las etiquetas de los alumnos en el eje X no se encimen utilizando rotación de etiquetas.

import pandas as pd
import matplotlib.pyplot as plt

# Cargar los datos del archivo Excel
file_path = 'calificaciones_alumnos.xlsx'
df = pd.read_excel(file_path)

# Verificar la estructura del DataFrame
print(df.head())

# Crear una figura y ejes para el gráfico
fig, ax = plt.subplots()

# Graficar las calificaciones
for column in df.columns[1:]: # Saltar la primera columna que es 'Nombre'
    ax.plot(df['Nombre'], df[column], marker='o', label=column)

# Configurar etiquetas y título del gráfico
ax.set_xlabel('Nombre del Alumno')
ax.set_ylabel('Calificación')
ax.set_title('Calificaciones de los Alumnos')
ax.legend()

# Rotar las etiquetas del eje X para evitar que se encimen
plt.xticks(rotation=45, ha='right')

# Ajustar el diseño para evitar recortes
plt.tight_layout()
```

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

Explicación del código

1. Importar las bibliotecas necesarias:

python

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

Se importan las bibliotecas `pandas` y `matplotlib`.

2. Cargar los datos del archivo Excel:

python

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

Se carga el archivo Excel en un DataFrame de `pandas`.

3. Verificar la estructura del DataFrame:

python

```
print(df.head())
```

Se imprime la cabecera del DataFrame para asegurarse de que los datos se han cargado correctamente.

4. Crear una figura y ejes para el gráfico:

python

```
fig, ax = plt.subplots()
```

Se crea una figura y un objeto de ejes para el gráfico.

5. Graficar las calificaciones:

python

```
for column in df.columns[1:]:
```

```
ax.plot(df['Nombre'], df[column], marker='o', label=column)
```

Se grafican las calificaciones de cada alumno para cada asignatura, usando la primera columna como etiquetas del eje X y las siguientes columnas como valores del eje Y.

6. Configurar etiquetas y título del gráfico:

python

```
ax.set_xlabel('Nombre del Alumno')
ax.set_ylabel('Calificación')
ax.set_title('Calificaciones de los Alumnos')
ax.legend()
```

Se configuran las etiquetas y el título del gráfico y se añade una leyenda.

7. Rotar las etiquetas del eje X:

python

```
plt.xticks(rotation=45, ha='right')
```

Se rotan las etiquetas del eje X 45 grados hacia la derecha para evitar que se encimen.

8. Ajustar el diseño del gráfico:

python

```
plt.tight_layout()
```

Se ajusta el diseño para evitar que las etiquetas o elementos del gráfico se corten.

9. Mostrar el gráfico:

python

```
plt.show()
```

Se muestra el gráfico.

Guarda este código en un archivo Python y ejecútalo para visualizar las calificaciones de los alumnos. Asegúrate de que el archivo "calificaciones_alumnos.xlsx" esté en el mismo directorio que tu script de Python.



como importo las bibliotecas anteriores



Para importar las bibliotecas `pandas` y `matplotlib`, necesitas asegurarte de que estén instaladas en tu entorno de Python. Si no las tienes instaladas, puedes instalarlas utilizando `pip`. Aquí tienes los comandos para instalarlas:

```
bash
```

```
pip install pandas matplotlib
```

Una vez instaladas, puedes importarlas en tu script de Python utilizando las siguientes líneas de código:

```
python
```

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

Pasos detallados:

1. Instalar las bibliotecas:

Abre tu terminal (o la consola en tu entorno de desarrollo integrado, como PyCharm) y ejecuta los siguientes comandos:

```
bash
```

```
pip install pandas
pip install matplotlib
```

2. Importar las bibliotecas en tu script de Python:

Al comienzo de tu archivo Python, añade las siguientes líneas:

```
python
```

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

Código completo con las importaciones:

python

```
# Este programa carga un archivo Excel con calificaciones de alumnos y grafica dichas calificaciones.
# Se asegura de que las etiquetas de los alumnos en el eje X no se encimen utilizando rotación de etiquetas.

import pandas as pd
import matplotlib.pyplot as plt

# Cargar los datos del archivo Excel
file_path = 'calificaciones_alumnos.xlsx'
df = pd.read_excel(file_path)

# Verificar la estructura del DataFrame
print(df.head())

# Crear una figura y ejes para el gráfico
fig, ax = plt.subplots()

# Graficar las calificaciones
for column in df.columns[1:]: # Saltar la primera columna que es 'Nombre'
    ax.plot(df['Nombre'], df[column], marker='o', label=column)

# Configurar etiquetas y título del gráfico
ax.set_xlabel('Nombre del Alumno')
ax.set_ylabel('Calificación')
ax.set_title('Calificaciones de los Alumnos')
ax.legend()

# Rotar las etiquetas del eje X para evitar que se encimen
plt.xticks(rotation=45, ha='right')

# Ajustar el diseño para evitar recortes
plt.tight_layout()

# Mostrar el gráfico
plt.show()
```

Guarda este código en un archivo Python, asegúrate de que el archivo "calificaciones_alumnos.xlsx" esté en el mismo directorio que tu script de Python, y ejecútalo. Esto cargará los datos desde el archivo Excel y generará el gráfico de calificaciones de los alumnos.



Dime por que me da errores tu codigo, este son los errores

```
"C:\Users\marco\OneDrive\Desktop\Curso Python 2024\codigos\M2.1 Automatizando Exce\pythonProjectM2.1 Automatizando Exce\venv\Scripts\python.exe"
"C:\Users\marco\OneDrive\Desktop\Curso Python 2024\codigos\M2.1 Automatizando Exce\pythonProjectM2.1 Automatizando Exce\GraficaAlumnos.py"
Traceback (most recent call last):
  File "C:\Users\marco\OneDrive\Desktop\Curso Python 2024\codigos\M2.1 Automatizando Exce\pythonProjectM2.1 Automatizando Exce\venv\lib\site-packages\pandas\compat\_optional.py", line 135, in import_optional_dependency
    module = importlib.import_module(name)
```

```
File "C:\Users\marco\AppData\Local\Programs\Python\Python310\lib\importlib\_init_.py", line
126, in import_module
    return _bootstrap._gcd_import(name[level:], package, level)
File "<frozen importlib._bootstrap>", line 1050, in _gcd_import
File "<frozen importlib._bootstrap>", line 1027, in _find_and_load
File "<frozen importlib._bootstrap>", line 1004, in _find_and_load_unlocked
ModuleNotFoundError: No module named 'openpyxl'
```

During handling of the above exception, another exception occurred:

Traceback (most recent call last):

```
File "C:\Users\marco\OneDrive\Desktop\Curso Python 2024\codigos\M2.1 Automatizando
Exce\pythonProjectM2.1 Automatizando Exce\GraficaAlumnos.py", line 9, in <module>
    df = pd.read_excel(file_path)
File "C:\Users\marco\OneDrive\Desktop\Curso Python 2024\codigos\M2.1 Automatizando
Exce\pythonProjectM2.1 Automatizando Exce\.venv\lib\site-packages\pandas\io\excel\_base.py",
line 495, in read_excel
    io = ExcelFile(
File "C:\Users\marco\OneDrive\Desktop\Curso Python 2024\codigos\M2.1 Automatizando
Exce\pythonProjectM2.1 Automatizando Exce\.venv\lib\site-packages\pandas\io\excel\_base.py",
line 1567, in __init__
    self._reader = self._engines[engine](
File "C:\Users\marco\OneDrive\Desktop\Curso Python 2024\codigos\M2.1 Automatizando
Exce\pythonProjectM2.1 Automatizando Exce\.venv\lib\site-
packages\pandas\io\excel\_openpyxl.py", line 552, in __init__
    import_optional_dependency("openpyxl")
File "C:\Users\marco\OneDrive\Desktop\Curso Python 2024\codigos\M2.1 Automatizando
Exce\pythonProjectM2.1 Automatizando Exce\.venv\lib\site-
packages\pandas\compat\_optional.py", line 138, in import_optional_dependency
    raise ImportError(msg)
ImportError: Missing optional dependency 'openpyxl'. Use pip or conda to install openpyxl.
```

Process finished with exit code 1



El error indica que la biblioteca `openpyxl`, que es necesaria para leer archivos Excel, no está instalada en tu entorno de Python. Puedes solucionar este problema instalando `openpyxl`.

Para instalar `openpyxl`, sigue estos pasos:

1. Abre tu terminal (o la consola en tu entorno de desarrollo integrado, como PyCharm).
2. Ejecuta el siguiente comando:

```
bash
```

```
pip install openpyxl
```

Esto instalará la biblioteca `openpyxl`.

Código completo con instrucciones para instalar bibliotecas:

Aquí tienes el código completo con una nota sobre la instalación de las bibliotecas necesarias:

```
python
```

```
# Este programa carga un archivo Excel con calificaciones de alumnos y grafica dichas calificaciones.
# Se asegura de que las etiquetas de los alumnos en el eje X no se encimen utilizando rotación de etiquetas.

# Primero, asegúrate de instalar las bibliotecas necesarias ejecutando estos comandos en tu terminal:
# pip install pandas matplotlib openpyxl

import pandas as pd
import matplotlib.pyplot as plt

# Cargar los datos del archivo Excel
file_path = 'calificaciones_alumnos.xlsx'
df = pd.read_excel(file_path)

# Verificar la estructura del DataFrame
print(df.head())

# Crear una figura y ejes para el gráfico
fig, ax = plt.subplots()

# Graficar las calificaciones
for column in df.columns[1:]: # Saltar la primera columna que es 'Nombre'
    ax.plot(df['Nombre'], df[column], marker='o', label=column)

# Configurar etiquetas y título del gráfico
ax.set_xlabel('Nombre del Alumno')
ax.set_ylabel('Calificación')
ax.set_title('Calificaciones de los Alumnos')
ax.legend()

# Rotar las etiquetas del eje X para evitar que se encimen
plt.xticks(rotation=45, ha='right')

# Ajustar el diseño para evitar recortes
plt.tight_layout()

# Mostrar el gráfico
plt.show()
```

Una vez que hayas instalado `openpyxl`, guarda este código en tu archivo Python y ejecútalo nuevamente. Deberías poder cargar los datos desde el archivo Excel y generar el gráfico de calificaciones de los alumnos sin problemas.

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 los datos de un archivo Excel llamado "calificaciones_alumnos.xlsx", agrega una columna llamada "Mat_Fisica" con valores aleatorios entre 0 y 100 con un decimal, y guarda el archivo actualizado. El programa utiliza las bibliotecas `pandas` para la manipulación de datos y `numpy` para generar valores aleatorios.

python

```
# Este programa carga un archivo Excel con calificaciones de alumnos, agrega una 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 los datos del archivo Excel
file_path = 'calificaciones_alumnos.xlsx'
df = pd.read_excel(file_path)

# Verificar la estructura del DataFrame
print(df.head())

# Generar valores aleatorios entre 0 y 100 con un decimal para la nueva columna
# 'Mat_Fisica'
df['Mat_Fisica'] = np.round(np.random.uniform(0, 100, len(df)), 1)

# Verificar la estructura del DataFrame después de agregar la nueva columna
print(df.head())

# Guardar el DataFrame actualizado en un nuevo archivo Excel
df.to_excel('calificaciones_alumnos_actualizado.xlsx', index=False)

# Verificar que el archivo se haya guardado correctamente
print("Archivo actualizado guardado como 'calificaciones_alumnos_actualizado.xlsx'")
```

Explicación del código

1. Importar las bibliotecas necesarias:

python

```
import pandas as pd
```



```
import numpy as np
```

Se importan las bibliotecas `pandas` y `numpy`.

2. Cargar los datos del archivo Excel:

```
python
```

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

Se carga el archivo Excel en un DataFrame de `pandas`.

3. Verificar la estructura del DataFrame:

```
python
```

```
print(df.head())
```

Se imprime la cabecera del DataFrame para asegurarse de que los datos se han cargado correctamente.

4. Generar valores aleatorios entre 0 y 100 con un decimal para la nueva columna 'Mat_Fisica':

```
python
```

```
df['Mat_Fisica'] = np.round(np.random.uniform(0, 100, len(df)), 1)
```

Se genera una nueva columna 'Mat_Fisica' con valores aleatorios entre 0 y 100, redondeados a un decimal, y se añade al DataFrame.

5. Verificar la estructura del DataFrame después de agregar la nueva columna:

```
python
```

```
print(df.head())
```

Se imprime la cabecera del DataFrame nuevamente para verificar que la nueva columna se haya agregado correctamente.

6. Guardar el DataFrame actualizado en un nuevo archivo Excel:

```
python
```

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

Se guarda el DataFrame actualizado en un nuevo archivo Excel llamado 'calificaciones_alumnos_actualizado.xlsx'.

7. Verificar que el archivo se haya guardado correctamente:

```
python
```

```
print("Archivo actualizado guardado como 'calificaciones_alumnos_actualizado.xlsx'")
```

Se imprime un mensaje confirmando que el archivo actualizado se ha guardado correctamente.

Guarda este código en un archivo Python y ejecútalo. Asegúrate de que el archivo "calificaciones_alumnos.xlsx" esté en el mismo directorio que tu script de Python. Este programa añadirá la columna "Mat_Fisica" con valores aleatorios y guardará el DataFrame actualizado 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 un programa en Python que carga los datos de un archivo Excel llamado "calificaciones_alumnos.xlsx", agrega una 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. El programa utiliza las bibliotecas `pandas` para la manipulación de datos y `numpy` para generar valores aleatorios.

```
python
```

```
# Este programa carga un archivo Excel con calificaciones de alumnos, agrega una 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 los datos del archivo Excel
file_path = 'calificaciones_alumnos.xlsx'
df = pd.read_excel(file_path)

# Verificar la estructura del DataFrame
print(df.head())

# Generar valores aleatorios entre 0 y 100 con un decimal para la nueva columna
'Mat_Fisica'
```

```
df['Mat_Fisica'] = np.round(np.random.uniform(0, 100, len(df)), 1)

# Verificar la estructura del DataFrame después de agregar la nueva columna
print(df.head())

# Ordenar la tabla por el campo 'Nombre'
df = df.sort_values(by='Nombre')

# Verificar la estructura del DataFrame después de ordenar
print(df.head())

# Guardar el DataFrame actualizado en un nuevo archivo Excel
df.to_excel('calificaciones_alumnos_actualizado.xlsx', index=False)

# Verificar que el archivo se haya guardado correctamente
print("Archivo actualizado guardado como 'calificaciones_alumnos_actualizado.xlsx'")
```

Explicación del código

1. Importar las bibliotecas necesarias:

python

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

Se importan las bibliotecas `pandas` y `numpy`.

2. Cargar los datos del archivo Excel:

python

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

Se carga el archivo Excel en un DataFrame de `pandas`.

3. Verificar la estructura del DataFrame:

python

```
print(df.head())
```

Se imprime la cabecera del DataFrame para asegurarse de que los datos se han cargado correctamente.

4. Generar valores aleatorios entre 0 y 100 con un decimal para la nueva columna 'Mat_Fisica':

python

```
df['Mat_Fisica'] = np.round(np.random.uniform(0, 100, len(df)), 1)
```

Se genera una nueva columna 'Mat_Fisica' con valores aleatorios entre 0 y 100, redondeados a un decimal, y se añade al DataFrame.

5. Verificar la estructura del DataFrame después de agregar la nueva columna:

python

```
print(df.head())
```

Se imprime la cabecera del DataFrame nuevamente para verificar que la nueva columna se haya agregado correctamente.

6. Ordenar la tabla por el campo 'Nombre':

python

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

Se ordena el DataFrame por el campo 'Nombre'.

7. Verificar la estructura del DataFrame después de ordenar:

python

```
print(df.head())
```

Se imprime la cabecera del DataFrame nuevamente para verificar que los datos se hayan ordenado correctamente.

8. Guardar el DataFrame actualizado en un nuevo archivo Excel:

python

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

Se guarda el DataFrame actualizado en un nuevo archivo Excel llamado 'calificaciones_alumnos_actualizado.xlsx'.

9. Verificar que el archivo se haya guardado correctamente:

python

```
print("Archivo actualizado guardado como 'calificaciones_alumnos_actualizado.xlsx'")
```

Se imprime un mensaje confirmando que el archivo actualizado se ha guardado correctamente.

Guarda este código en un archivo Python y ejecútalo. Asegúrate de que el archivo "calificaciones_alumnos.xlsx" esté en el mismo directorio que tu script de Python. Este programa añadirá la columna "Mat_Fisica" con valores aleatorios, ordenará el DataFrame por el campo "Nombre", y guardará el DataFrame actualizado 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 un programa en Python que carga los datos de un archivo Excel llamado "calificaciones_alumnos.xlsx", agrega una 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. Además, el programa imprime el número de registros (filas) y campos (columnas) en la tabla. El programa utiliza las bibliotecas `pandas` para la manipulación de datos y `numpy` para generar valores aleatorios.

python

```
# Este programa carga un archivo Excel con calificaciones de alumnos, agrega una columna
# llamada 'Mat_Fisica'
# con valores aleatorios entre 0 y 100 con un decimal, ordena la tabla por el campo
# 'Nombre',
# imprime el número de registros y campos en la tabla, y guarda el archivo actualizado.

import pandas as pd
import numpy as np

# Cargar los datos del archivo Excel
file_path = 'calificaciones_alumnos.xlsx'
df = pd.read_excel(file_path)

# Verificar la estructura del DataFrame
print(df.head())

# Imprimir el número de registros (filas) y campos (columnas)
num_registros = df.shape[0]
num_campos = df.shape[1]
print(f'Número de registros: {num_registros}')
print(f'Número de campos: {num_campos}')

# Generar valores aleatorios entre 0 y 100 con un decimal para la nueva columna
'Mat_Fisica'
```

```
df['Mat_Fisica'] = np.round(np.random.uniform(0, 100, len(df)), 1)

# Verificar la estructura del DataFrame después de agregar la nueva columna
print(df.head())

# Ordenar la tabla por el campo 'Nombre'
df = df.sort_values(by='Nombre')

# Verificar la estructura del DataFrame después de ordenar
print(df.head())

# Guardar el DataFrame actualizado en un nuevo archivo Excel
df.to_excel('calificaciones_alumnos_actualizado.xlsx', index=False)

# Verificar que el archivo se haya guardado correctamente
print("Archivo actualizado guardado como 'calificaciones_alumnos_actualizado.xlsx'")
```

Explicación del código

1. Importar las bibliotecas necesarias:

python

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

Se importan las bibliotecas `pandas` y `numpy`.

2. Cargar los datos del archivo Excel:

python

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

Se carga el archivo Excel en un DataFrame de `pandas`.

3. Verificar la estructura del DataFrame:

python

```
print(df.head())
```

Se imprime la cabecera del DataFrame para asegurarse de que los datos se han cargado correctamente.

4. Imprimir el número de registros (filas) y campos (columnas):

python

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

Se calculan e imprimen el número de registros (filas) y campos (columnas) en el DataFrame.

5. Generar valores aleatorios entre 0 y 100 con un decimal para la nueva columna 'Mat_Fisica':

python

```
df['Mat_Fisica'] = np.round(np.random.uniform(0, 100, len(df)), 1)
```

Se genera una nueva columna 'Mat_Fisica' con valores aleatorios entre 0 y 100, redondeados a un decimal, y se añade al DataFrame.

6. Verificar la estructura del DataFrame después de agregar la nueva columna:

python

```
print(df.head())
```

Se imprime la cabecera del DataFrame nuevamente para verificar que la nueva columna se haya agregado correctamente.

7. Ordenar la tabla por el campo 'Nombre':

python

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

Se ordena el DataFrame por el campo 'Nombre'.

8. Verificar la estructura del DataFrame después de ordenar:

python

```
print(df.head())
```

Se imprime la cabecera del DataFrame nuevamente para verificar que los datos se hayan ordenado correctamente.

9. Guardar el DataFrame actualizado en un nuevo archivo Excel:

```
python
```

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

Se guarda el DataFrame actualizado en un nuevo archivo Excel llamado 'calificaciones_alumnos_actualizado.xlsx'.

10. Verificar que el archivo se haya guardado correctamente:

```
python
```

```
print("Archivo actualizado guardado como 'calificaciones_alumnos_actualizado.xlsx'")
```

Se imprime un mensaje confirmando que el archivo actualizado se ha guardado correctamente.

Guarda este código en un archivo Python y ejecútalo. Asegúrate de que el archivo "calificaciones_alumnos.xlsx" esté en el mismo directorio que tu script de Python. Este programa añadirá la columna "Mat_Fisica" con valores aleatorios, ordenará el DataFrame por el campo "Nombre", imprimirá el número de registros y campos, y guardará el DataFrame actualizado 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 un programa en Python que carga los datos de un archivo Excel llamado "calificaciones_alumnos.xlsx", agrega una columna llamada "Mat_Fisica" con valores aleatorios entre 0 y 100 con un decimal, ordena la tabla por el campo "Nombre", imprime el número de registros y campos, identifica qué campos son numéricos, y guarda el archivo actualizado. El programa utiliza las bibliotecas `pandas` para la manipulación de datos y `numpy` para generar valores aleatorios.

```
python
```

```
# Este programa carga un archivo Excel con calificaciones de alumnos, agrega una columna
# llamada 'Mat_Fisica'
# con valores aleatorios entre 0 y 100 con un decimal, ordena la tabla por el campo
# 'Nombre',
# imprime el número de registros y campos en la tabla, identifica qué campos son
# numéricos,
# y guarda el archivo actualizado.
```

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



```
# Cargar los datos del archivo Excel
file_path = 'calificaciones_alumnos.xlsx'
df = pd.read_excel(file_path)

# Verificar la estructura del DataFrame
print(df.head())

# Imprimir el número de registros (filas) y campos (columnas)
num_registros = df.shape[0]
num_campos = df.shape[1]
print(f'Número de registros: {num_registros}')
print(f'Número de campos: {num_campos}')

# Identificar los campos numéricos
campos_numericos = df.select_dtypes(include=[np.number]).columns.tolist()
print(f'Campos numéricos: {campos_numericos}')

# Generar valores aleatorios entre 0 y 100 con un decimal para la nueva columna
'Mat_Fisica'
df['Mat_Fisica'] = np.round(np.random.uniform(0, 100, len(df)), 1)

# Verificar la estructura del DataFrame después de agregar la nueva columna
print(df.head())

# Ordenar la tabla por el campo 'Nombre'
df = df.sort_values(by='Nombre')

# Verificar la estructura del DataFrame después de ordenar
print(df.head())

# Guardar el DataFrame actualizado en un nuevo archivo Excel
df.to_excel('calificaciones_alumnos_actualizado.xlsx', index=False)

# Verificar que el archivo se haya guardado correctamente
print("Archivo actualizado guardado como 'calificaciones_alumnos_actualizado.xlsx'")
```

Explicación del código

1. Importar las bibliotecas necesarias:

python

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

Se importan las bibliotecas `pandas` y `numpy`.

2. Cargar los datos del archivo Excel:

python

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

Se carga el archivo Excel en un DataFrame de `pandas`.

3. Verificar la estructura del DataFrame:

```
python
```

```
print(df.head())
```

Se imprime la cabecera del DataFrame para asegurarse de que los datos se han cargado correctamente.

4. Imprimir el número de registros (filas) y campos (columnas):

```
python
```

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

```
print(f'Número de campos: {num_campos}')
```

Se calculan e imprimen el número de registros (filas) y campos (columnas) en el DataFrame.

5. Identificar los campos numéricos:

```
python
```

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

Se identifican e imprimen los campos numéricos en el DataFrame.

6. Generar valores aleatorios entre 0 y 100 con un decimal para la nueva columna 'Mat_Fisica':

```
python
```

```
df['Mat_Fisica'] = np.round(np.random.uniform(0, 100, len(df)), 1)
```

Se genera una nueva columna 'Mat_Fisica' con valores aleatorios entre 0 y 100, redondeados a un decimal, y se añade al DataFrame.

7. Verificar la estructura del DataFrame después de agregar la nueva columna:

```
python
```

```
print(df.head())
```

Se imprime la cabecera del DataFrame nuevamente para verificar que la nueva columna se haya agregado correctamente.

8. Ordenar la tabla por el campo 'Nombre':

python

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

Se ordena el DataFrame por el campo 'Nombre'.

9. Verificar la estructura del DataFrame después de ordenar:

python

```
print(df.head())
```

Se imprime la cabecera del DataFrame nuevamente para verificar que los datos se hayan ordenado correctamente.

10. Guardar el DataFrame actualizado en un nuevo archivo Excel:

python

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

Se guarda el DataFrame actualizado en un nuevo archivo Excel llamado 'calificaciones_alumnos_actualizado.xlsx'.

11. Verificar que el archivo se haya guardado correctamente:

python

```
print("Archivo actualizado guardado como 'calificaciones_alumnos_actualizado.xlsx'")
```

Se imprime un mensaje confirmando que el archivo actualizado se ha guardado correctamente.

Guarda este código en un archivo Python y ejecútalo. Asegúrate de que el archivo "calificaciones_alumnos.xlsx" esté en el mismo directorio que tu script de Python. Este programa añadirá la columna "Mat_Fisica" con valores aleatorios, ordenará el DataFrame por el campo "Nombre", imprimirá el número de registros y campos, identificará los campos numéricos, y guardará el DataFrame actualizado en un nuevo archivo Excel.



Quiero que me generes 15 programas diferentes en python, utilizando de la tabla los campos "Mat_CalculoIntegral", "Mat_ProgramacionOOP", "Mat_EstructuraDatos" y "Mat_Fisica", estos

campos representan las calificaciones de materias y como dato las calificaciones aprobatorias están entre 70 a 100 y menores a 70 son reprobatorias, al inicio de cada código agrega un comentario explicando lo que hace el programa y numerando el programa de acuerdo a los 15 programas pedidos y documenta las líneas importantes del código



Aquí tienes 15 programas en Python, cada uno con una funcionalidad diferente utilizando los campos "Mat_CalculoIntegral", "Mat_ProgramacionOOP", "Mat_EstructuraDatos" y "Mat_Fisica" para manipular y analizar las calificaciones. Cada programa está numerado y comentado adecuadamente.

Programa 1: Calcular el promedio de calificaciones de cada alumno

python

```
# Programa 1: Este programa calcula el promedio de calificaciones de cada alumno
# utilizando los campos "Mat_CalculoIntegral", "Mat_ProgramacionOOP",
"Mat_EstructuraDatos" y "Mat_Fisica".

import pandas as pd

# Cargar los datos del archivo Excel
file_path = 'calificaciones_alumnos.xlsx'
df = pd.read_excel(file_path)

# Calcular el promedio de calificaciones de cada alumno
df['Promedio'] = df[['Mat_CalculoIntegral', 'Mat_ProgramacionOOP', 'Mat_EstructuraDatos',
'Mat_Fisica']].mean(axis=1)

# Verificar la estructura del DataFrame después de agregar la nueva columna
print(df.head())

# Guardar el DataFrame actualizado en un nuevo archivo Excel
df.to_excel('calificaciones_promedio.xlsx', index=False)
```

Programa 2: Contar el número de materias aprobadas por cada alumno

python

```
# Programa 2: Este programa cuenta el número de materias aprobadas por cada alumno
# utilizando los campos "Mat_CalculoIntegral", "Mat_ProgramacionOOP",
"Mat_EstructuraDatos" y "Mat_Fisica".

import pandas as pd

# Cargar los datos del archivo Excel
file_path = 'calificaciones_alumnos.xlsx'
df = pd.read_excel(file_path)

# Contar el número de materias aprobadas por cada alumno
df['Materias_Aprobadas'] = (df[['Mat_CalculoIntegral', 'Mat_ProgramacionOOP',
'Mat_EstructuraDatos', 'Mat_Fisica']] >= 70).sum(axis=1)

# Verificar la estructura del DataFrame después de agregar la nueva columna
```

```
print(df.head())

# Guardar el DataFrame actualizado en un nuevo archivo Excel
df.to_excel('calificaciones_materias_aprobadas.xlsx', index=False)
```

Programa 3: Contar el número de materias reprobadas por cada alumno

python

```
# Programa 3: Este programa cuenta el número de materias reprobadas por cada alumno
# utilizando los campos "Mat_CalculoIntegral", "Mat_ProgramacionOOP",
"Mat_EstructuraDatos" y "Mat_Fisica".

import pandas as pd

# Cargar los datos del archivo Excel
file_path = 'calificaciones_alumnos.xlsx'
df = pd.read_excel(file_path)

# Contar el número de materias reprobadas por cada alumno
df['Materias_Reprobadas'] = (df[['Mat_CalculoIntegral', 'Mat_ProgramacionOOP',
'Mat_EstructuraDatos', 'Mat_Fisica']] < 70).sum(axis=1)

# Verificar la estructura del DataFrame después de agregar la nueva columna
print(df.head())

# Guardar el DataFrame actualizado en un nuevo archivo Excel
df.to_excel('calificaciones_materias_reprobadas.xlsx', index=False)
```

Programa 4: Identificar alumnos con promedio mayor a 85

python

```
# Programa 4: Este programa identifica los alumnos con un promedio mayor a 85
# utilizando los campos "Mat_CalculoIntegral", "Mat_ProgramacionOOP",
"Mat_EstructuraDatos" y "Mat_Fisica".

import pandas as pd

# Cargar los datos del archivo Excel
file_path = 'calificaciones_alumnos.xlsx'
df = pd.read_excel(file_path)

# Calcular el promedio de calificaciones de cada alumno
df['Promedio'] = df[['Mat_CalculoIntegral', 'Mat_ProgramacionOOP', 'Mat_EstructuraDatos',
'Mat_Fisica']].mean(axis=1)

# Identificar los alumnos con un promedio mayor a 85
df_alumnos_destacados = df[df['Promedio'] > 85]

# Verificar la estructura del DataFrame de alumnos destacados
print(df_alumnos_destacados)

# Guardar el DataFrame de alumnos destacados en un nuevo archivo Excel
df_alumnos_destacados.to_excel('alumnos_destacados.xlsx', index=False)
```

Programa 5: Identificar alumnos que reprobaron todas las materias

python

```
# Programa 5: Este programa identifica los alumnos que reprobaron todas las materias
# utilizando los campos "Mat_CalculoIntegral", "Mat_Programacion00P",
# "Mat_EstructuraDatos" y "Mat_Fisica".

import pandas as pd

# Cargar los datos del archivo Excel
file_path = 'calificaciones_alumnos.xlsx'
df = pd.read_excel(file_path)

# Identificar los alumnos que reprobaron todas las materias
df_alumnos_reprobados = df[(df['Mat_CalculoIntegral'] < 70) & (df['Mat_Programacion00P'] < 70) & (df['Mat_EstructuraDatos'] < 70) & (df['Mat_Fisica'] < 70)]

# Verificar la estructura del DataFrame de alumnos reprobados
print(df_alumnos_reprobados)

# Guardar el DataFrame de alumnos reprobados en un nuevo archivo Excel
df_alumnos_reprobados.to_excel('alumnos_reprobados.xlsx', index=False)
```

Programa 6: Calcular la media de las calificaciones de cada materia

python

```
# Programa 6: Este programa calcula la media de las calificaciones de cada materia
# utilizando los campos "Mat_CalculoIntegral", "Mat_Programacion00P",
# "Mat_EstructuraDatos" y "Mat_Fisica".

import pandas as pd

# Cargar los datos del archivo Excel
file_path = 'calificaciones_alumnos.xlsx'
df = pd.read_excel(file_path)

# Calcular la media de las calificaciones de cada materia
media_calificaciones = df[['Mat_CalculoIntegral', 'Mat_Programacion00P',
'Mat_EstructuraDatos', 'Mat_Fisica']].mean()

# Verificar la media de las calificaciones
print(media_calificaciones)

# Guardar la media de las calificaciones en un archivo Excel
media_calificaciones.to_excel('media_calificaciones.xlsx')
```

Programa 7: Calcular la mediana de las calificaciones de cada materia

python

```
# Programa 7: Este programa calcula la mediana de las calificaciones de cada materia
# utilizando los campos "Mat_CalculoIntegral", "Mat_Programacion00P",
# "Mat_EstructuraDatos" y "Mat_Fisica".
```

```
import pandas as pd

# Cargar los datos del archivo Excel
file_path = 'calificaciones_alumnos.xlsx'
df = pd.read_excel(file_path)

# Calcular la mediana de las calificaciones de cada materia
mediana_calificaciones = df[['Mat_CalculoIntegral', 'Mat_Programacion00P',
'Mat_EstructuraDatos', 'Mat_Fisica']].median()

# Verificar la mediana de las calificaciones
print(mediana_calificaciones)

# Guardar la mediana de las calificaciones en un archivo Excel
mediana_calificaciones.to_excel('mediana_calificaciones.xlsx')
```

Programa 8: Calcular la desviación estándar de las calificaciones de cada materia

python

```
# Programa 8: Este programa calcula la desviación estándar de las calificaciones de cada
materia
# utilizando los campos "Mat_CalculoIntegral", "Mat_Programacion00P",
"Mat_EstructuraDatos" y "Mat_Fisica".

import pandas as pd

# Cargar los datos del archivo Excel
file_path = 'calificaciones_alumnos.xlsx'
df = pd.read_excel(file_path)

# Calcular la desviación estándar de las calificaciones de cada materia
desviacion_estandar_calificaciones = df[['Mat_CalculoIntegral', 'Mat_Programacion00P',
'Mat_EstructuraDatos', 'Mat_Fisica']].std()

# Verificar la desviación estándar de las calificaciones
print(desviacion_estandar_calificaciones)

# Guardar la desviación estándar de las calificaciones en un archivo Excel
desviacion_estandar_calificaciones.to_excel('desviacion_estandar_calificaciones.xlsx')
```

Programa 9: Identificar la calificación más alta y más baja de cada materia

python

```
# Programa 9: Este programa identifica la calificación más alta y más baja de cada materia
# utilizando los campos "Mat_CalculoIntegral", "Mat_Programacion00P",
"Mat_EstructuraDatos" y "Mat_Fisica".

import pandas as pd

# Cargar los datos del archivo Excel
file_path = 'calificaciones_alumnos.xlsx'
df = pd.read_excel(file_path)

# Identificar la calificación más alta y más baja de cada materia
max_calificaciones = df[['Mat_CalculoIntegral', 'Mat_Programacion00P',
```

```

'Mat_EstructuraDatos', 'Mat_Fisica']]).max()
min_calificaciones = df[['Mat_CalculoIntegral', 'Mat_ProgramacionOOP',
'Mat_EstructuraDatos', 'Mat_Fisica']]).min()

# Verificar la calificación más alta y más baja de cada materia
print(f'Calificaciones más altas:\n{max_calificaciones}')
print(f'Calificaciones más bajas:\n{min_calificaciones}')

# Guardar las calificaciones más altas y más bajas en un archivo Excel
pd.DataFrame({'Maximas': max_calificaciones, 'Minimas':
min_calificaciones}).to_excel('max_min_calificaciones.xlsx')

```

Programa 10: Crear un histograma de las calificaciones de "Mat_Fisica"

python

```

# Programa 10: Este programa crea un histograma de las calificaciones de "Mat_Fisica"
# utilizando los campos "Mat_CalculoIntegral", "Mat_ProgramacionOOP",
"Mat_EstructuraDatos" y "Mat_Fisica".

import pandas as pd
import matplotlib.pyplot as plt

# Cargar los datos del archivo Excel
file_path = 'calificaciones_alumnos.xlsx'
df = pd.read_excel(file_path)

# Crear un histograma de las calificaciones de "Mat_Fisica"
plt.hist(df['Mat_Fisica'], bins=10, edgecolor='black')
plt.title('Histograma de Calificaciones de Mat_Fisica')
plt.xlabel('Calificación')
plt.ylabel('Frecuencia')
plt.grid(True)

# Mostrar el histograma
plt.show()

```

Programa 11: Crear un boxplot de las calificaciones de todas las materias

python

```

# Programa 11: Este programa crea un boxplot de las calificaciones de todas las materias
# utilizando los campos "Mat_CalculoIntegral", "Mat_ProgramacionOOP",
"Mat_EstructuraDatos" y "Mat_Fisica".

import pandas as pd
import matplotlib.pyplot as plt

# Cargar los datos del archivo Excel
file_path = 'calificaciones_alumnos.xlsx'
df = pd.read_excel(file_path)

# Crear un boxplot de las calificaciones de todas las materias
df[['Mat_CalculoIntegral', 'Mat_ProgramacionOOP', 'Mat_EstructuraDatos',
'Mat_Fisica']].boxplot()
plt.title('Boxplot de Calificaciones de Todas las Materias')
plt.xlabel('Materia')
plt.ylabel('Calificación')

```



```
plt.grid(True)

# Mostrar el boxplot
plt.show()
```

Programa 12: Calcular el porcentaje de aprobados y reprobados por materia

python

```
# Programa 12: Este programa calcula el porcentaje de aprobados y reprobados por materia
# utilizando los campos "Mat_CalculoIntegral", "Mat_Programacion00P",
"Mat_EstructuraDatos" y "Mat_Fisica".

import pandas as pd

# Cargar los datos del archivo Excel
file_path = 'calificaciones_alumnos.xlsx'
df = pd.read_excel(file_path)

# Calcular el porcentaje de aprobados y reprobados por materia
porcentaje_aprobados = (df[['Mat_CalculoIntegral', 'Mat_Programacion00P',
'Mat_EstructuraDatos', 'Mat_Fisica']] >= 70).mean() * 100
porcentaje_reprobados = (df[['Mat_CalculoIntegral', 'Mat_Programacion00P',
'Mat_EstructuraDatos', 'Mat_Fisica']] < 70).mean() * 100

# Verificar el porcentaje de aprobados y reprobados por materia
print(f'Porcentaje de Aprobados:\n{porcentaje_aprobados}')
print(f'Porcentaje de Reprobados:\n{porcentaje_reprobados}')

# Guardar el porcentaje de aprobados y reprobados en un archivo Excel
pd.DataFrame({'Aprobados': porcentaje_aprobados, 'Reprobados':
porcentaje_reprobados}).to_excel('porcentaje_aprobados_reprobados.xlsx')
```

Programa 13: Crear una tabla pivot para promedio de calificaciones por alumno y materia

python

```
# Programa 13: Este programa crea una tabla pivot que muestra el promedio de
calificaciones por alumno y materia
# utilizando los campos "Mat_CalculoIntegral", "Mat_Programacion00P",
"Mat_EstructuraDatos" y "Mat_Fisica".

import pandas as pd

# Cargar los datos del archivo Excel
file_path = 'calificaciones_alumnos.xlsx'
df = pd.read_excel(file_path)

# Crear una tabla pivot para promedio de calificaciones por alumno y materia
pivot_table = pd.pivot_table(df, values=['Mat_CalculoIntegral', 'Mat_Programacion00P',
'Mat_EstructuraDatos', 'Mat_Fisica'], index='Nombre', aggfunc='mean')

# Verificar la tabla pivot
print(pivot_table)
```

```
# Guardar la tabla pivot en un archivo Excel
pivot_table.to_excel('tabla_pivot_promedio_calificaciones.xlsx')
```

Programa 14: Crear una tabla que muestra los alumnos con calificaciones sobresalientes (mayores a 90) en cualquier materia

python

```
# Programa 14: Este programa crea una tabla que muestra los alumnos con calificaciones
sobresalientes (mayores a 90) en cualquier materia
# utilizando los campos "Mat_CalculoIntegral", "Mat_ProgramacionOOP",
"Mat_EstructuraDatos" y "Mat_Fisica".

import pandas as pd

# Cargar los datos del archivo Excel
file_path = 'calificaciones_alumnos.xlsx'
df = pd.read_excel(file_path)

# Crear una tabla que muestra los alumnos con calificaciones sobresalientes en cualquier
materia
alumnos_sobresalientes = df[(df['Mat_CalculoIntegral'] > 90) | (df['Mat_ProgramacionOOP']
> 90) | (df['Mat_EstructuraDatos'] > 90) | (df['Mat_Fisica'] > 90)]

# Verificar la tabla de alumnos sobresalientes
print(alumnos_sobresalientes)

# Guardar la tabla de alumnos sobresalientes en un archivo Excel
alumnos_sobresalientes.to_excel('alumnos_sobresalientes.xlsx', index=False)
```

Programa 15: Crear un gráfico de barras apiladas mostrando la distribución de aprobados y reprobados por materia

python

```
# Programa 15: Este programa crea un gráfico de barras apiladas mostrando la distribución
de aprobados y reprobados por materia
# utilizando los campos "Mat_CalculoIntegral", "Mat_ProgramacionOOP",
"Mat_EstructuraDatos" y "Mat_Fisica".

import pandas as pd
import matplotlib.pyplot as plt

# Cargar los datos del archivo Excel
file_path = 'calificaciones_alumnos.xlsx'
df = pd.read_excel(file_path)

# Contar el número de aprobados y reprobados por materia
aprobados = (df[['Mat_CalculoIntegral', 'Mat_ProgramacionOOP', 'Mat_EstructuraDatos',
'Mat_Fisica']] >= 70).sum()
reprobados = (df[['Mat_CalculoIntegral', 'Mat_ProgramacionOOP', 'Mat_EstructuraDatos',
'Mat_Fisica']] < 70).sum()

# Crear un gráfico de barras apiladas
fig, ax = plt.subplots()
bar_width = 0.5
```

```

index = range(len(aprobados))

bar1 = ax.bar(index, aprobados, bar_width, label='Aprobados')
bar2 = ax.bar(index, reprobados, bar_width, bottom=aprobados, label='Reprobados')

ax.set_xlabel('Materia')
ax.set_ylabel('Número de Alumnos')
ax.set_title('Distribución de Aprobados y Reprobados por Materia')
ax.set_xticks(index)
ax.set_xticklabels(['Calculo Integral', 'Programación OOP', 'Estructura de Datos',
                    'Física'])
ax.legend()

# Mostrar el gráfico
plt.show()

```

Guarda cada uno de estos programas en archivos Python individuales y ejecútalos según sea necesario. Cada programa realiza una tarea específica utilizando los campos de calificaciones y está debidamente documentado. Asegúrate de tener el archivo "calificaciones_alumnos.xlsx" en el mismo directorio que tus scripts de Python.

ChatGPT puede cometer errores. Considera verificar la información importante.