

## M2.1 Automatizando 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



Claro, a continuación te muestro un programa en Python que carga el archivo "calificaciones\_alumnos.xlsx", extrae las calificaciones de las materias y crea un gráfico para cada alumno sin encimar las etiquetas en el eje X. El código incluye comentarios explicativos:

python

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

# Este programa carga un archivo Excel con calificaciones de alumnos,
# extrae las calificaciones de tres materias y grafica las calificaciones
# de cada alumno sin encimar las etiquetas en el eje X.

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

# Extraer los nombres de los alumnos y las calificaciones
nombres = df['Nombre']
calificaciones = df[['Mat_CalculoIntegral', 'Mat_ProgramacionOOP', 'Mat_EstructuraDatos']]

# Crear una lista de materias
materias = ['Cálculo Integral', 'Programación OOP', 'Estructura de Datos']

# Configurar el tamaño de la figura para que las etiquetas no se encimen
plt.figure(figsize=(10, 6))

# Graficar las calificaciones de cada alumno
for i, nombre in enumerate(nombres):
    plt.plot(materias, calificaciones.iloc[i], marker='o', label=nombre)

# Configurar etiquetas y título del gráfico
plt.xlabel('Materias')
plt.ylabel('Calificaciones')
plt.title('Calificaciones de Alumnos por Materia')

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

# Mostrar leyenda con los nombres de los alumnos
plt.legend(loc='best', bbox_to_anchor=(1, 1), title='Alumnos')
```

```
# Ajustar los márgenes para que las etiquetas de la leyenda no se recorten
plt.tight_layout()

# Mostrar el gráfico
plt.show()
```

## Explicación del código

### 1. Importar las librerías necesarias:

python

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

### 2. Cargar el archivo Excel:

python

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

### 3. Extraer los nombres de los alumnos y las calificaciones:

python

```
nombres = df['Nombre']
calificaciones = df[['Mat_CalculoIntegral', 'Mat_Programacion00P',
'Mat_EstructuraDatos']]
```

### 4. Crear una lista de materias:

python

```
materias = ['Cálculo Integral', 'Programación 00P', 'Estructura de Datos']
```

### 5. Configurar el tamaño de la figura:

python

```
plt.figure(figsize=(10, 6))
```

## 6. Graficar las calificaciones de cada alumno:

python

```
for i, nombre in enumerate(nombres):
    plt.plot(materias, calificaciones.iloc[i], marker='o', label=nombre)
```

## 7. Configurar etiquetas y título del gráfico:

python

```
plt.xlabel('Materias')
plt.ylabel('Calificaciones')
plt.title('Calificaciones de Alumnos por Materia')
```

## 8. Rotar las etiquetas del eje X para evitar que se encimen:

python

```
plt.xticks(rotation=45)
```

## 9. Mostrar la leyenda con los nombres de los alumnos:

python

```
plt.legend(loc='best', bbox_to_anchor=(1, 1), title='Alumnos')
```

## 10. Ajustar los márgenes:

python

```
plt.tight_layout()
```

## 11. Mostrar el gráfico:

python

```
plt.show()
```

Este código debería proporcionarte un gráfico claro y legible de las calificaciones de los alumnos para las tres materias especificadas.

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



Claro, a continuación te muestro un programa en Python que carga el archivo "calificaciones\_alumnos.xlsx", agrega una columna llamada "Mat\_Fisica" con valores aleatorios entre 0 y 100 con un decimal, y guarda el archivo modificado. El código incluye comentarios explicativos:

python

```
import pandas as pd
import numpy as np

# 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 modificado.

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

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

# Guardar el archivo Excel modificado
df.to_excel('calificaciones_alumnos_modificado.xlsx', index=False)

# Mostrar un mensaje indicando que el archivo se ha guardado exitosamente
print('El archivo modificado ha sido guardado como
"calificaciones_alumnos_modificado.xlsx"')
```

## Explicación del código

### 1. Importar las librerías necesarias:

python

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

### 2. Cargar el archivo Excel:

python

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

### 3. Generar valores aleatorios entre 0 y 100 con un decimal para la columna "Mat\_Fisica":

python

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

- `np.random.uniform(0, 100, size=len(df))`: Genera valores aleatorios entre 0 y 100 para cada fila del DataFrame.
- `np.round(..., 1)`: Redondea los valores generados a un decimal.

### 4. Guardar el archivo Excel modificado:

python

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

### 5. Mostrar un mensaje indicando que el archivo se ha guardado exitosamente:

python

```
print('El archivo modificado ha sido guardado como  
"calificaciones_alumnos_modificado.xlsx"')
```

Este código agregará la nueva columna "Mat\_Fisica" con valores aleatorios a tu archivo Excel y guardará el archivo modificado con un nuevo nombre.



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



Claro, aquí tienes el programa en Python que carga el archivo "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 modificado. El código incluye comentarios explicativos:

python

```
import pandas as pd
import numpy as np

# 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 modificado.

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

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

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

# Guardar el archivo Excel modificado
df_ordenado.to_excel('calificaciones_alumnos_modificado.xlsx', index=False)

# Mostrar un mensaje indicando que el archivo se ha guardado exitosamente
print('El archivo modificado ha sido guardado como "calificaciones_alumnos_modificado.xlsx"')
```

## Explicación del código

### 1. Importar las librerías necesarias:

python

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

### 2. Cargar el archivo Excel:

python

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

### 3. Generar valores aleatorios entre 0 y 100 con un decimal para la columna "Mat\_Fisica":

python

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

- ``np.random.uniform(0, 100, size=len(df))``: Genera valores aleatorios entre 0 y 100 para cada fila del DataFrame.
- ``np.round(..., 1)``: Redondea los valores generados a un decimal.

#### 4. Ordenar la tabla por el campo "Nombre":

python

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

- ``df.sort_values(by='Nombre')``: Ordena el DataFrame por la columna "Nombre".

#### 5. Guardar el archivo Excel modificado:

python

```
df_ordenado.to_excel('calificaciones_alumnos_modificado.xlsx', index=False)
```

#### 6. Mostrar un mensaje indicando que el archivo se ha guardado exitosamente:

python

```
print('El archivo modificado ha sido guardado como  
"calificaciones_alumnos_modificado.xlsx"')
```

Este código agregará la nueva columna "Mat\_Fisica" con valores aleatorios a tu archivo Excel, ordenará la tabla por el campo "Nombre" y guardará el archivo modificado con un nuevo nombre.



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



Claro, a continuación te muestro un programa en Python que carga el archivo "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", guarda el archivo modificado y muestra el número de registros y campos de la tabla. El código incluye comentarios explicativos:

python

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

```
# 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", guarda el archivo modificado,
# y muestra el número de registros y campos de la tabla.

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

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

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

# Guardar el archivo Excel modificado
df_ordenado.to_excel('calificaciones_alumnos_modificado.xlsx', index=False)

# Obtener el número de registros y campos de la tabla
num_registros = df_ordenado.shape[0]
num_campos = df_ordenado.shape[1]

# Mostrar el número de registros y campos
print(f'El número de registros en la tabla es: {num_registros}')
print(f'El número de campos en la tabla es: {num_campos}')

# Mostrar un mensaje indicando que el archivo se ha guardado exitosamente
print('El archivo modificado ha sido guardado como
"calificaciones_alumnos_modificado.xlsx"')
```

## Explicación del código

### 1. Importar las librerías necesarias:

python

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

### 2. Cargar el archivo Excel:

python

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

### 3. Generar valores aleatorios entre 0 y 100 con un decimal para la columna "Mat\_Fisica":

python

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



- ``np.random.uniform(0, 100, size=len(df))``: Genera valores aleatorios entre 0 y 100 para cada fila del DataFrame.
- ``np.round(..., 1)``: Redondea los valores generados a un decimal.

#### 4. Ordenar la tabla por el campo "Nombre":

python

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

- ``df.sort_values(by='Nombre')``: Ordena el DataFrame por la columna "Nombre".

#### 5. Guardar el archivo Excel modificado:

python

```
df_ordenado.to_excel('calificaciones_alumnos_modificado.xlsx', index=False)
```

#### 6. Obtener el número de registros y campos de la tabla:

python

```
num_registros = df_ordenado.shape[0]
num_campos = df_ordenado.shape[1]
```

- ``df_ordenado.shape[0]``: Obtiene el número de registros (filas) del DataFrame.
- ``df_ordenado.shape[1]``: Obtiene el número de campos (columnas) del DataFrame.

#### 7. Mostrar el número de registros y campos:

python

```
print(f'El número de registros en la tabla es: {num_registros}')
print(f'El número de campos en la tabla es: {num_campos}')
```

#### 8. Mostrar un mensaje indicando que el archivo se ha guardado exitosamente:

python

```
print('El archivo modificado ha sido guardado como  
"calificaciones_alumnos_modificado.xlsx"')
```

Este código agregará la nueva columna "Mat\_Fisica" con valores aleatorios a tu archivo Excel, ordenará la tabla por el campo "Nombre", guardará el archivo modificado con un nuevo nombre y mostrará el número de registros y campos de la tabla.

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



Claro, aquí tienes un programa en Python que carga el archivo "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", guarda el archivo modificado, muestra el número de registros y campos de la tabla, y muestra qué campos son numéricos. El código incluye comentarios explicativos:

python

```
import pandas as pd
import numpy as np

# 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", guarda el archivo modificado,
# muestra el número de registros y campos de la tabla, y muestra qué campos son numéricos.

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

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

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

# Guardar el archivo Excel modificado
df_ordenado.to_excel('calificaciones_alumnos_modificado.xlsx', index=False)

# Obtener el número de registros y campos de la tabla
num_registros = df_ordenado.shape[0]
num_campos = df_ordenado.shape[1]

# Mostrar el número de registros y campos
print(f'El número de registros en la tabla es: {num_registros}')
print(f'El número de campos en la tabla es: {num_campos}')

# Identificar los campos numéricos
campos_numericos = df_ordenado.select_dtypes(include=[np.number]).columns.tolist()

# Mostrar los campos numéricos
print(f'Los campos numéricos en la tabla son: {campos_numericos}')

# Mostrar un mensaje indicando que el archivo se ha guardado exitosamente
print('El archivo modificado ha sido guardado como "calificaciones_alumnos_modificado.xlsx"')
```

## Explicación del código

### 1. Importar las librerías necesarias:

python

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

## 2. Cargar el archivo Excel:

python

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

## 3. Generar valores aleatorios entre 0 y 100 con un decimal para la columna "Mat\_Fisica":

python

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

- `np.random.uniform(0, 100, size=len(df))`: Genera valores aleatorios entre 0 y 100 para cada fila del DataFrame.
- `np.round(..., 1)`: Redondea los valores generados a un decimal.

## 4. Ordenar la tabla por el campo "Nombre":

python

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

- `df.sort_values(by='Nombre')`: Ordena el DataFrame por la columna "Nombre".

## 5. Guardar el archivo Excel modificado:

python

```
df_ordenado.to_excel('calificaciones_alumnos_modificado.xlsx', index=False)
```

## 6. Obtener el número de registros y campos de la tabla:

python

```
num_registros = df_ordenado.shape[0]
num_campos = df_ordenado.shape[1]
```

- `df_ordenado.shape[0]`: Obtiene el número de registros (filas) del DataFrame.
- `df_ordenado.shape[1]`: Obtiene el número de campos (columnas) del DataFrame.

## 7. Mostrar el número de registros y campos:

python

```
print(f'El número de registros en la tabla es: {num_registros}')
print(f'El número de campos en la tabla es: {num_campos}')
```

## 8. Identificar los campos numéricos:

python

```
campos_numericos = df_ordenado.select_dtypes(include=[np.number]).columns.tolist()
```

- `df_ordenado.select_dtypes(include=[np.number])`: Selecciona solo las columnas de tipo numérico del DataFrame.
- `.columns.tolist()`: Convierte los nombres de las columnas seleccionadas a una lista.

## 9. Mostrar los campos numéricos:

python

```
print(f'Los campos numéricos en la tabla son: {campos_numericos}')
```

## 10. Mostrar un mensaje indicando que el archivo se ha guardado exitosamente:

python

```
print('El archivo modificado ha sido guardado como  
"calificaciones_alumnos_modificado.xlsx"')
```

Este código agregará la nueva columna "Mat\_Fisica" con valores aleatorios a tu archivo Excel, ordenará la tabla por el campo "Nombre", guardará el archivo modificado con un nuevo nombre, mostrará el número de registros y campos de la tabla, y mostrará qué campos son numéricos.



Quiero saber el promedio de la columna Promedio, al inicio del código agrega un comentario explicando lo que hace el programa y documenta las líneas importantes del código



Claro, aquí tienes un programa en Python que carga el archivo "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", guarda el archivo modificado, muestra el número de registros y campos de la tabla, muestra qué campos son numéricos, y calcula el promedio de la columna "Promedio". El código incluye comentarios explicativos:

python

```
import pandas as pd
import numpy as np

# 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", guarda el archivo modificado,
# muestra el número de registros y campos de la tabla, muestra qué campos son numéricos,
# y calcula el promedio de la columna "Promedio".

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

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

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

# Guardar el archivo Excel modificado
df_ordenado.to_excel('calificaciones_alumnos_modificado.xlsx', index=False)

# Obtener el número de registros y campos de la tabla
num_registros = df_ordenado.shape[0]
num_campos = df_ordenado.shape[1]

# Mostrar el número de registros y campos
print(f'El número de registros en la tabla es: {num_registros}')
print(f'El número de campos en la tabla es: {num_campos}')

# Identificar los campos numéricos
campos_numericos = df_ordenado.select_dtypes(include=[np.number]).columns.tolist()

# Mostrar los campos numéricos
print(f'Los campos numéricos en la tabla son: {campos_numericos}')

# Calcular el promedio de la columna "Promedio"
if 'Promedio' in df_ordenado.columns:
    promedio = df_ordenado['Promedio'].mean()
    print(f'El promedio de la columna "Promedio" es: {promedio}')
else:
    print('La columna "Promedio" no existe en la tabla.')

# Mostrar un mensaje indicando que el archivo se ha guardado exitosamente
print('El archivo modificado ha sido guardado como "calificaciones_alumnos_modificado.xlsx"')
```

## Explicación del código

### 1. Importar las librerías necesarias:

python

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

## 2. Cargar el archivo Excel:

python

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

## 3. Generar valores aleatorios entre 0 y 100 con un decimal para la columna "Mat\_Fisica":

python

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

- `np.random.uniform(0, 100, size=len(df))`: Genera valores aleatorios entre 0 y 100 para cada fila del DataFrame.
- `np.round(..., 1)`: Redondea los valores generados a un decimal.

## 4. Ordenar la tabla por el campo "Nombre":

python

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

- `df.sort_values(by='Nombre')`: Ordena el DataFrame por la columna "Nombre".

## 5. Guardar el archivo Excel modificado:

python

```
df_ordenado.to_excel('calificaciones_alumnos_modificado.xlsx', index=False)
```

## 6. Obtener el número de registros y campos de la tabla:

python

```
num_registros = df_ordenado.shape[0]
num_campos = df_ordenado.shape[1]
```

- `df_ordenado.shape[0]`: Obtiene el número de registros (filas) del DataFrame.
- `df_ordenado.shape[1]`: Obtiene el número de campos (columnas) del DataFrame.

## 7. Mostrar el número de registros y campos:

python

```
print(f'El número de registros en la tabla es: {num_registros}')
print(f'El número de campos en la tabla es: {num_campos}')
```

## 8. Identificar los campos numéricos:

python

```
campos_numericos = df_ordenado.select_dtypes(include=[np.number]).columns.tolist()
```

- `df_ordenado.select_dtypes(include=[np.number])`: Selecciona solo las columnas de tipo numérico del DataFrame.
- `.columns.tolist()`: Convierte los nombres de las columnas seleccionadas a una lista.

## 9. Mostrar los campos numéricos:

python

```
print(f'Los campos numéricos en la tabla son: {campos_numericos}')
```

## 10. Calcular el promedio de la columna "Promedio":

python

```
if 'Promedio' in df_ordenado.columns:
    promedio = df_ordenado['Promedio'].mean()
    print(f'El promedio de la columna "Promedio" es: {promedio}')
else:
    print('La columna "Promedio" no existe en la tabla.')
```

- `df_ordenado['Promedio'].mean()`: Calcula el promedio de los valores en la columna "Promedio".
- La instrucción `if 'Promedio' in df_ordenado.columns` verifica si la columna "Promedio" existe en el DataFrame.

## 11. Mostrar un mensaje indicando que el archivo se ha guardado exitosamente:

python

```
print('El archivo modificado ha sido guardado como')
```

```
"calificaciones_alumnos_modificado.xlsx")
```

Este código agregará la nueva columna "Mat\_Fisica" con valores aleatorios a tu archivo Excel, ordenará la tabla por el campo "Nombre", guardará el archivo modificado con un nuevo nombre, mostrará el número de registros y campos de la tabla, mostrará qué campos son numéricos y calculará el promedio de la columna "Promedio".



Agregar la columna "Mat\_Estatus" en la cual se deberá insertar el texto "Aprobada" o "Reprobada" con la regla: si el valor de la columna "Promedio" en el mismo renglón es mayor o igual 70.0 agregar "Aprobada", en caso contrario agregar "Reprobada"



Claro, aquí tienes un programa en Python que carga el archivo "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", agrega la columna "Mat\_Estatus" basada en el valor de la columna "Promedio", guarda el archivo modificado, muestra el número de registros y campos de la tabla, muestra qué campos son numéricos, y calcula el promedio de la columna "Promedio". El código incluye comentarios explicativos:

python

```
import pandas as pd
import numpy as np

# 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", agrega la columna "Mat_Estatus"
# basada en el valor de la columna "Promedio", guarda el archivo modificado,
# muestra el número de registros y campos de la tabla, muestra qué campos son numéricos,
# y calcula el promedio de la columna "Promedio".

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

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

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

# Agregar la columna "Mat_Estatus" basada en la columna "Promedio"
df_ordenado['Mat_Estatus'] = df_ordenado['Promedio'].apply(lambda x: 'Aprobada' if x >= 70
else 'Reprobada')

# Guardar el archivo Excel modificado
df_ordenado.to_excel('calificaciones_alumnos_modificado.xlsx', index=False)

# Obtener el número de registros y campos de la tabla
num_registros = df_ordenado.shape[0]
```



```
num_campos = df_ordenado.shape[1]

# Mostrar el número de registros y campos
print(f'El número de registros en la tabla es: {num_registros}')
print(f'El número de campos en la tabla es: {num_campos}')

# Identificar los campos numéricos
campos_numericos = df_ordenado.select_dtypes(include=[np.number]).columns.tolist()

# Mostrar los campos numéricos
print(f'Los campos numéricos en la tabla son: {campos_numericos}')

# Calcular el promedio de la columna "Promedio"
if 'Promedio' in df_ordenado.columns:
    promedio = df_ordenado['Promedio'].mean()
    print(f'El promedio de la columna "Promedio" es: {promedio}')
else:
    print('La columna "Promedio" no existe en la tabla.')

# Mostrar un mensaje indicando que el archivo se ha guardado exitosamente
print('El archivo modificado ha sido guardado como "calificaciones_alumnos_modificado.xlsx"')
```

## Explicación del código

### 1. Importar las librerías necesarias:

python

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

### 2. Cargar el archivo Excel:

python

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

### 3. Generar valores aleatorios entre 0 y 100 con un decimal para la columna "Mat\_Fisica":

python

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

- `np.random.uniform(0, 100, size=len(df))`: Genera valores aleatorios entre 0 y 100 para cada fila del DataFrame.
- `np.round(..., 1)`: Redondea los valores generados a un decimal.

### 4. Ordenar la tabla por el campo "Nombre":

python

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

- `df.sort_values(by='Nombre')`: Ordena el DataFrame por la columna "Nombre".

## 5. Agregar la columna "Mat\_Estatus" basada en la columna "Promedio":

python

```
df_ordenado['Mat_Estatus'] = df_ordenado['Promedio'].apply(lambda x: 'Aprobada' if x >= 70 else 'Reprobada')
```

- `df_ordenado['Promedio'].apply(lambda x: 'Aprobada' if x >= 70 else 'Reprobada')`: Aplica una función lambda para asignar "Aprobada" si el valor de "Promedio" es mayor o igual a 70, y "Reprobada" en caso contrario.

## 6. Guardar el archivo Excel modificado:

python

```
df_ordenado.to_excel('calificaciones_alumnos_modificado.xlsx', index=False)
```

## 7. Obtener el número de registros y campos de la tabla:

python

```
num_registros = df_ordenado.shape[0]
num_campos = df_ordenado.shape[1]
```

- `df_ordenado.shape[0]`: Obtiene el número de registros (filas) del DataFrame.
- `df_ordenado.shape[1]`: Obtiene el número de campos (columnas) del DataFrame.

## 8. Mostrar el número de registros y campos:

python

```
print(f'El número de registros en la tabla es: {num_registros}')
print(f'El número de campos en la tabla es: {num_campos}')
```

## 9. Identificar los campos numéricos:

python

```
campos_numericos = df_ordenado.select_dtypes(include=[np.number]).columns.tolist()
```

- `df_ordenado.select_dtypes(include=[np.number])`: Selecciona solo las columnas de tipo numérico del DataFrame.
- `.columns.tolist()`: Convierte los nombres de las columnas seleccionadas a una lista.

## 10. Mostrar los campos numéricos:

python

```
print(f'Los campos numéricos en la tabla son: {campos_numericos}')
```

## 11. Calcular el promedio de la columna "Promedio":

python

```
if 'Promedio' in df_ordenado.columns:
    promedio = df_ordenado['Promedio'].mean()
    print(f'El promedio de la columna "Promedio" es: {promedio}')
```

```
else:
    print('La columna "Promedio" no existe en la tabla.')
```

- `df_ordenado['Promedio'].mean()`: Calcula el promedio de los valores en la columna "Promedio".
- La instrucción `if 'Promedio' in df_ordenado.columns` verifica si la columna "Promedio" existe en el DataFrame.

## 12. Mostrar un mensaje indicando que el archivo se ha guardado exitosamente:

python

```
print('El archivo modificado ha sido guardado como  
"calificaciones_alumnos_modificado.xlsx"')
```

Este código agregará la nueva columna "Mat\_Fisica" con valores aleatorios a tu archivo Excel, ordenará la tabla por el campo "Nombre", agregará la columna "Mat\_Estatus" basada en el valor de la columna "Promedio", guardará el archivo modificado con un nuevo nombre, mostrará el número de registros y campos de la tabla, mostrará qué campos son numéricos y calculará el promedio de la columna "Promedio".



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 el color de fondo en verde para las celdas de la columna Promedio si el valor es mayor o igual a 70.0, en caso contrario, agregar el color de fondo de la celda en color rojo; al inicio del código agrega un comentario explicando lo que hace el programa y documenta las líneas importantes del código



Claro, aquí tienes un programa en Python que carga el archivo "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", agrega la columna "Mat\_Estatus" basada en el valor de la columna "Promedio", guarda el archivo modificado, muestra el número de registros y campos de la tabla, muestra qué campos son numéricos, calcula el promedio de la columna "Promedio", y colorea las celdas de la columna "Promedio" en verde si el valor es mayor o igual a 70, y en rojo en caso contrario. El código incluye comentarios explicativos:

python

```
import pandas as pd
import numpy as np
from openpyxl import load_workbook
from openpyxl.styles import PatternFill

# 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", agrega la columna "Mat_Estatus"
# basada en el valor de la columna "Promedio", guarda el archivo modificado,
# muestra el número de registros y campos de la tabla, muestra qué campos son numéricos,
# calcula el promedio de la columna "Promedio", y colorea las celdas de la columna
# "Promedio"
# en verde si el valor es mayor o igual a 70, y en rojo en caso contrario.

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

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

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

# Agregar la columna "Mat_Estatus" basada en la columna "Promedio"
df_ordenado['Mat_Estatus'] = df_ordenado['Promedio'].apply(lambda x: 'Aprobada' if x >= 70
else 'Reprobada')

# Guardar el archivo Excel modificado
archivo_modificado = 'calificaciones_alumnos_modificado.xlsx'
df_ordenado.to_excel(archivo_modificado, index=False)

# Colorear las celdas de la columna "Promedio" en verde si el valor es mayor o igual a 70,
# y en rojo en caso contrario
wb = load_workbook(archivo_modificado)
```

```
ws = wb.active

# Definir los colores
verde = PatternFill(start_color='00FF00', end_color='00FF00', fill_type='solid')
rojo = PatternFill(start_color='FF0000', end_color='FF0000', fill_type='solid')

# Aplicar los colores a las celdas de la columna "Promedio"
for row in ws.iter_rows(min_row=2, min_col=ws.max_column, max_col=ws.max_column):
    for cell in row:
        if cell.value >= 70:
            cell.fill = verde
        else:
            cell.fill = rojo

# Guardar el archivo Excel con las celdas coloreadas
wb.save(archivo_modificado)

# Obtener el número de registros y campos de la tabla
num_registros = df_ordenado.shape[0]
num_campos = df_ordenado.shape[1]

# Mostrar el número de registros y campos
print(f'El número de registros en la tabla es: {num_registros}')
print(f'El número de campos en la tabla es: {num_campos}')

# Identificar los campos numéricos
campos_numericos = df_ordenado.select_dtypes(include=[np.number]).columns.tolist()

# Mostrar los campos numéricos
print(f'Los campos numéricos en la tabla son: {campos_numericos}')

# Calcular el promedio de la columna "Promedio"
if 'Promedio' in df_ordenado.columns:
    promedio = df_ordenado['Promedio'].mean()
    print(f'El promedio de la columna "Promedio" es: {promedio}')
else:
    print('La columna "Promedio" no existe en la tabla.')

# Mostrar un mensaje indicando que el archivo se ha guardado exitosamente
print(f'El archivo modificado ha sido guardado como "{archivo_modificado}"')
```

## Explicación del código

### 1. Importar las librerías necesarias:

python

```
import pandas as pd
import numpy as np
from openpyxl import load_workbook
from openpyxl.styles import PatternFill
```

### 2. Cargar el archivo Excel:

python

```
archivo_excel = 'calificaciones_alumnos.xlsx'
```

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

### 3. Generar valores aleatorios entre 0 y 100 con un decimal para la columna "Mat\_Fisica":

python

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

- `np.random.uniform(0, 100, size=len(df))`: Genera valores aleatorios entre 0 y 100 para cada fila del DataFrame.
- `np.round(..., 1)`: Redondea los valores generados a un decimal.

### 4. Ordenar la tabla por el campo "Nombre":

python

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

- `df.sort_values(by='Nombre')`: Ordena el DataFrame por la columna "Nombre".

### 5. Agregar la columna "Mat\_Estatus" basada en la columna "Promedio":

python

```
df_ordenado['Mat_Estatus'] = df_ordenado['Promedio'].apply(lambda x: 'Aprobada' if x >= 70 else 'Reprobada')
```

- `df_ordenado['Promedio'].apply(lambda x: 'Aprobada' if x >= 70 else 'Reprobada')`: Aplica una función lambda para asignar "Aprobada" si el valor de "Promedio" es mayor o igual a 70, y "Reprobada" en caso contrario.

### 6. Guardar el archivo Excel modificado:

python

```
archivo_modificado = 'calificaciones_alumnos_modificado.xlsx'
df_ordenado.to_excel(archivo_modificado, index=False)
```

### 7. Colorear las celdas de la columna "Promedio" en verde si el valor es mayor o igual a 70, y en rojo en caso contrario:

python

```
wb = load_workbook(archivo_modificado)
ws = wb.active

# Definir los colores
```

```
verde = PatternFill(start_color='00FF00', end_color='00FF00', fill_type='solid')
rojo = PatternFill(start_color='FF0000', end_color='FF0000', fill_type='solid')

# Aplicar los colores a las celdas de la columna "Promedio"
for row in ws.iter_rows(min_row=2, min_col=ws.max_column, max_col=ws.max_column):
    for cell in row:
        if cell.value >= 70:
            cell.fill = verde
        else:
            cell.fill = rojo

# Guardar el archivo Excel con las celdas coloreadas
wb.save(archivo_modificado)
```

- `PatternFill(start_color='00FF00', end_color='00FF00', fill_type='solid')`: Define el color de relleno verde.
- `PatternFill(start_color='FF0000', end_color='FF0000', fill_type='solid')`: Define el color de relleno rojo.
- `ws.iter_rows(min_row=2, min_col=ws.max_column, max_col=ws.max_column)`: Itera sobre las celdas de la columna "Promedio" (ajustando `min_col` y `max_col` según sea necesario).
- `cell.fill = verde` o `cell.fill = rojo`: Asigna el color de relleno a la celda basada en el valor.

## 8. Obtener el número de registros y campos de la tabla:

python

```
num_registros = df_ordenado.shape[0]
num_campos = df_ordenado.shape[1]
```

- `df_ordenado.shape[0]`: Obtiene el número de registros (filas) del DataFrame.
- `df_ordenado.shape[1]`: Obtiene el número de campos (columnas) del DataFrame.

## 9. Mostrar el número de registros y campos:

python

```
print(f'El número de registros en la tabla es: {num_registros}')
print(f'El número de campos en la tabla es: {num_campos}')
```

## 10. Identificar los campos numéricos:

python

```
campos_numericos = df_ordenado.select_dtypes(include=[np.number]).columns.tolist()
```

- `df_ordenado.select_dtypes(include=[np.number])`: Selecciona solo las columnas de tipo numérico del DataFrame.
- `.columns.tolist()`: Convierte los nombres de las columnas seleccionadas a una lista.

## 11. Mostrar los campos numéricos:

python

```
print(f'Los campos numéricos en la tabla son: {campos_numericos}')
```

## 12. Calcular el promedio de la columna "Promedio":

python

```
if 'Promedio' in df_ordenado.columns:
    promedio = df_ordenado['Promedio'].mean()
    print(f'El promedio de la columna "Promedio" es: {promedio}')
else:
    print('La columna "Promedio" no existe en la tabla.')
```

- `df_ordenado['Promedio'].mean()`: Calcula el promedio de los valores en la columna "Promedio".
- La instrucción `if 'Promedio' in df_ordenado.columns` verifica si la columna "Promedio" existe en el DataFrame.

## 13. Mostrar un mensaje indicando que el archivo se ha guardado exitosamente:

python

```
print(f'El archivo modificado ha sido guardado como "{archivo_modificado}")')
```

Este código realizará todas las tareas especificadas y coloreará las celdas de la columna "Promedio" de acuerdo a los valores.



del código anterior, no pedí que agregaras la columna "Mat\_Fisica", haz el ajuste



Entendido, aquí está el programa ajustado que carga el archivo "calificaciones\_alumnos.xlsx", ordena la tabla por el campo "Nombre", agrega la columna "Mat\_Estatus" basada en el valor de la columna "Promedio", guarda el archivo modificado, muestra el número de registros y campos de la tabla, muestra qué campos son numéricos, calcula el promedio de la columna "Promedio", y colorea las celdas de la columna "Promedio" en verde si el valor es mayor o igual a 70, y en rojo en caso contrario. El código incluye comentarios explicativos:



python

```
import pandas as pd
import numpy as np
from openpyxl import load_workbook
from openpyxl.styles import PatternFill

# Este programa carga un archivo Excel con calificaciones de alumnos,
# ordena la tabla por el campo "Nombre", agrega la columna "Mat_Estatus"
# basada en el valor de la columna "Promedio", guarda el archivo modificado,
# muestra el número de registros y campos de la tabla, muestra qué campos son numéricos,
# calcula el promedio de la columna "Promedio", y colorea las celdas de la columna
"Promedio"
# en verde si el valor es mayor o igual a 70, y en rojo en caso contrario.

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

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

# Agregar la columna "Mat_Estatus" basada en la columna "Promedio"
df_ordenado['Mat_Estatus'] = df_ordenado['Promedio'].apply(lambda x: 'Aprobada' si x >= 70
else 'Reprobada')

# Guardar el archivo Excel modificado
archivo_modificado = 'calificaciones_alumnos_modificado.xlsx'
df_ordenado.to_excel(archivo_modificado, index=False)

# Colorear las celdas de la columna "Promedio" en verde si el valor es mayor o igual a 70,
y en rojo en caso contrario
wb = load_workbook(archivo_modificado)
ws = wb.active

# Definir los colores
verde = PatternFill(start_color='00FF00', end_color='00FF00', fill_type='solid')
rojo = PatternFill(start_color='FF0000', end_color='FF0000', fill_type='solid')

# Aplicar los colores a las celdas de la columna "Promedio"
for row in ws.iter_rows(min_row=2, min_col=ws.max_column, max_col=ws.max_column):
    for cell in row:
        if cell.value >= 70:
            cell.fill = verde
        else:
            cell.fill = rojo

# Guardar el archivo Excel con las celdas coloreadas
wb.save(archivo_modificado)

# Obtener el número de registros y campos de la tabla
num_registros = df_ordenado.shape[0]
num_campos = df_ordenado.shape[1]

# Mostrar el número de registros y campos
print(f'El número de registros en la tabla es: {num_registros}')
print(f'El número de campos en la tabla es: {num_campos}')

# Identificar los campos numéricos
campos_numericos = df_ordenado.select_dtypes(include=[np.number]).columns.tolist()

# Mostrar los campos numéricos
print(f'Los campos numéricos en la tabla son: {campos_numericos}')

# Calcular el promedio de la columna "Promedio"
if 'Promedio' in df_ordenado.columns:
```

```
promedio = df_ordenado['Promedio'].mean()
print(f'El promedio de la columna "Promedio" es: {promedio}')
else:
    print('La columna "Promedio" no existe en la tabla.')

# Mostrar un mensaje indicando que el archivo se ha guardado exitosamente
print(f'El archivo modificado ha sido guardado como "{archivo_modificado}"')
```

## Explicación del código

### 1. Importar las librerías necesarias:

python

```
import pandas as pd
import numpy as np
from openpyxl import load_workbook
from openpyxl.styles import PatternFill
```

### 2. Cargar el archivo Excel:

python

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

### 3. Ordenar la tabla por el campo "Nombre":

python

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

- `df.sort_values(by='Nombre')`: Ordena el DataFrame por la columna "Nombre".

### 4. \*\*Agregar la columna "Mat\_Estatus"



solo aplica lo último que te pedí, no agregues ajustes anteriores



Aquí tienes el código ajustado para cargar el archivo "calificaciones\_alumnos.xlsx" y agregar el color de fondo en verde para las celdas de la columna "Promedio" si el valor es mayor o igual a 70, en caso contrario, agregar el color de fondo en rojo. El código incluye comentarios explicativos:

python

```
import pandas as pd
from openpyxl import load_workbook
from openpyxl.styles import PatternFill

# Este programa carga un archivo Excel con calificaciones de alumnos,
# colorea las celdas de la columna "Promedio" en verde si el valor es mayor o igual a 70,
# y en rojo en caso contrario.

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

# Guardar el archivo Excel modificado
archivo_modificado = 'calificaciones_alumnos_modificado.xlsx'
df.to_excel(archivo_modificado, index=False)

# Colorear las celdas de la columna "Promedio" en verde si el valor es mayor o igual a 70,
# y en rojo en caso contrario
wb = load_workbook(archivo_modificado)
ws = wb.active

# Definir los colores
verde = PatternFill(start_color='00FF00', end_color='00FF00', fill_type='solid')
rojo = PatternFill(start_color='FF0000', end_color='FF0000', fill_type='solid')

# Encontrar el índice de la columna "Promedio"
col_idx = None
for i, col in enumerate(ws.iter_cols(1, ws.max_column), start=1):
    if col[0].value == 'Promedio':
        col_idx = i
        break

# Aplicar los colores a las celdas de la columna "Promedio"
if col_idx:
    for cell in ws.iter_rows(min_row=2, min_col=col_idx, max_col=col_idx):
        for c in cell:
            if c.value >= 70:
                c.fill = verde
            else:
                c.fill = rojo

# Guardar el archivo Excel con las celdas coloreadas
wb.save(archivo_modificado)

print(f'El archivo modificado ha sido guardado como "{archivo_modificado}")')
```

## Explicación del código

### 1. Importar las librerías necesarias:

python

```
import pandas as pd
from openpyxl import load_workbook
from openpyxl.styles import PatternFill
```

## 2. Cargar el archivo Excel:

python

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

## 3. Guardar el archivo Excel modificado:

python

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

## 4. Colorear las celdas de la columna "Promedio" en verde si el valor es mayor o igual a 70, y en rojo en caso contrario:

python

```
wb = load_workbook(archivo_modificado)  
ws = wb.active  
  
# Definir los colores  
verde = PatternFill(start_color='00FF00', end_color='00FF00', fill_type='solid')  
rojo = PatternFill(start_color='FF0000', end_color='FF0000', fill_type='solid')  
  
# Encontrar el índice de la columna "Promedio"  
col_idx = None  
for i, col in enumerate(ws.iter_cols(1, ws.max_column), start=1):  
    if col[0].value == 'Promedio':  
        col_idx = i  
        break  
  
# Aplicar los colores a las celdas de la columna "Promedio"  
if col_idx:  
    for cell in ws.iter_rows(min_row=2, min_col=col_idx, max_col=col_idx):  
        for c in cell:  
            if c.value >= 70:  
                c.fill = verde  
            else:  
                c.fill = rojo  
  
# Guardar el archivo Excel con las celdas coloreadas  
wb.save(archivo_modificado)
```

## 5. Mostrar un mensaje indicando que el archivo se ha guardado exitosamente:

python

```
print(f'El archivo modificado ha sido guardado como "{archivo_modificado}")')
```



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 el formato negrita a la columna "Nombre"; al inicio del código agrega un comentario Solo aplica lo que te pedí, no agregues ajustes anteriores



Claro, aquí tienes un programa en Python que carga el archivo "calificaciones\_alumnos.xlsx" y agrega el formato de negrita a la columna "Nombre". El código incluye comentarios explicativos:

python

```
import pandas as pd
from openpyxl import load_workbook
from openpyxl.styles import Font

# Este programa carga un archivo Excel con calificaciones de alumnos,
# y agrega el formato de negrita a la columna "Nombre".

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

# Guardar el archivo Excel modificado
archivo_modificado = 'calificaciones_alumnos_modificado.xlsx'
df.to_excel(archivo_modificado, index=False)

# Cargar el archivo Excel para modificarlo con openpyxl
wb = load_workbook(archivo_modificado)
ws = wb.active

# Definir el formato de negrita
negrita = Font(bold=True)

# Encontrar el índice de la columna "Nombre"
col_idx = None
for i, col in enumerate(ws.iter_cols(1, ws.max_column), start=1):
    if col[0].value == 'Nombre':
        col_idx = i
        break

# Aplicar el formato de negrita a las celdas de la columna "Nombre"
if col_idx:
    for cell in ws.iter_rows(min_row=2, min_col=col_idx, max_col=col_idx):
        for c in cell:
            c.font = negrita

# Guardar el archivo Excel con el formato aplicado
wb.save(archivo_modificado)

print(f'El archivo modificado ha sido guardado como "{archivo_modificado}")')
```

## Explicación del código

### 1. Importar las librerías necesarias:

python

```
import pandas as pd
from openpyxl import load_workbook
from openpyxl.styles import Font
```

## 2. Cargar el archivo Excel:

python

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

## 3. Guardar el archivo Excel modificado:

python

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

## 4. Cargar el archivo Excel para modificarlo con openpyxl:

python

```
wb = load_workbook(archivo_modificado)
ws = wb.active
```

## 5. Definir el formato de negrita:

python

```
negrita = Font(bold=True)
```

## 6. Encontrar el índice de la columna "Nombre":

python

```
col_idx = None
for i, col in enumerate(ws.iter_cols(1, ws.max_column), start=1):
    if col[0].value == 'Nombre':
        col_idx = i
        break
```

## 7. Aplicar el formato de negrita a las celdas de la columna "Nombre":

python

```
if col_idx:
    for cell in ws.iter_rows(min_row=2, min_col=col_idx, max_col=col_idx):
        for c in cell:
            c.font = negrita
```

## 8. Guardar el archivo Excel con el formato aplicado:

python

```
wb.save(archivo_modificado)
```

## 9. Mostrar un mensaje indicando que el archivo se ha guardado exitosamente:

python

```
print(f'El archivo modificado ha sido guardado como "{archivo_modificado}")')
```



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 quitar los espacios a la izquierda y a la derecha en la columna "NombreAlumno"; al inicio del código agrega un comentario explicando lo que hace el programa y documenta las líneas importantes del código.

Solo aplica lo que te pedí, no agregues ajustes anteriores



Aquí tienes el programa en Python que carga el archivo "calificaciones\_alumnos.xlsx" y quita los espacios a la izquierda y a la derecha en la columna "NombreAlumno". El código incluye comentarios explicativos y solo aplica lo que solicitaste:

python

```
import pandas as pd

# Este programa carga un archivo Excel con calificaciones de alumnos,
# y quita los espacios a la izquierda y a la derecha en la columna "NombreAlumno".
```

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

# Quitar los espacios a la izquierda y a la derecha en la columna "NombreAlumno"
if 'NombreAlumno' in df.columns:
    df['NombreAlumno'] = df['NombreAlumno'].str.strip()

# Guardar el archivo Excel modificado
archivo_modificado = 'calificaciones_alumnos_modificado.xlsx'
df.to_excel(archivo_modificado, index=False)

print(f'El archivo modificado ha sido guardado como "{archivo_modificado}")')
```

## Explicación del código

### 1. Importar la librería necesaria:

python

```
import pandas as pd
```

### 2. Cargar el archivo Excel:

python

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

### 3. Quitar los espacios a la izquierda y a la derecha en la columna "NombreAlumno":

python

```
if 'NombreAlumno' in df.columns:
    df['NombreAlumno'] = df['NombreAlumno'].str.strip()
```

- `df['NombreAlumno'].str.strip()`: Elimina los espacios en blanco al principio y al final de cada valor en la columna "NombreAlumno".

### 4. Guardar el archivo Excel modificado:

python

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

### 5. Mostrar un mensaje indicando que el archivo se ha guardado exitosamente:



```
python
```

```
print(f'El archivo modificado ha sido guardado como "{archivo_modificado}')
```



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 quitar los espacios a la izquierda y a la derecha en la columna "ApellidoAlumno"; al inicio del código agrega un comentario explicando lo que hace el programa y documenta las líneas importantes del código.

Solo aplica lo que te pedí, no agregues ajustes anteriores



Aquí tienes el programa en Python que carga el archivo "calificaciones\_alumnos.xlsx" y quita los espacios a la izquierda y a la derecha en la columna "ApellidoAlumno". El código incluye comentarios explicativos y solo aplica lo que solicitaste:

```
python
```

```
import pandas as pd

# Este programa carga un archivo Excel con calificaciones de alumnos,
# y quita los espacios a la izquierda y a la derecha en la columna "ApellidoAlumno".

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

# Quitar los espacios a la izquierda y a la derecha en la columna "ApellidoAlumno"
if 'ApellidoAlumno' in df.columns:
    df['ApellidoAlumno'] = df['ApellidoAlumno'].str.strip()

# Guardar el archivo Excel modificado
archivo_modificado = 'calificaciones_alumnos_modificado.xlsx'
df.to_excel(archivo_modificado, index=False)

print(f'El archivo modificado ha sido guardado como "{archivo_modificado}')
```

## Explicación del código

### 1. Importar la librería necesaria:

python

```
import pandas as pd
```

## 2. Cargar el archivo Excel:

python

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

## 3. Quitar los espacios a la izquierda y a la derecha en la columna "ApellidoAlumno":

python

```
if 'ApellidoAlumno' in df.columns:
    df['ApellidoAlumno'] = df['ApellidoAlumno'].str.strip()
```

- `df['ApellidoAlumno'].str.strip()`: Elimina los espacios en blanco al principio y al final de cada valor en la columna "ApellidoAlumno".

## 4. Guardar el archivo Excel modificado:

python

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

## 5. Mostrar un mensaje indicando que el archivo se ha guardado exitosamente:

python

```
print(f'El archivo modificado ha sido guardado como "{archivo_modificado}")')
```



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 redondear a dos decimales la columna "Mat\_CalculoIntegral"; al inicio del código agrega un comentario explicando lo que hace el programa y documenta las líneas importantes del código. Solo aplica lo que te pedí, no agregues ajustes anteriores



Aquí tienes el programa en Python que carga el archivo "calificaciones\_alumnos.xlsx" y redondea a dos decimales los valores en la columna "Mat\_CalculoIntegral". El código incluye comentarios explicativos y solo aplica lo que solicitaste:

python

```
import pandas as pd

# Este programa carga un archivo Excel con calificaciones de alumnos,
# y redondea a dos decimales los valores en la columna "Mat_CalculoIntegral".

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

# Redondear a dos decimales la columna "Mat_CalculoIntegral"
if 'Mat_CalculoIntegral' in df.columns:
    df['Mat_CalculoIntegral'] = df['Mat_CalculoIntegral'].round(2)

# Guardar el archivo Excel modificado
archivo_modificado = 'calificaciones_alumnos_modificado.xlsx'
df.to_excel(archivo_modificado, index=False)

print(f'El archivo modificado ha sido guardado como "{archivo_modificado}")')
```

## Explicación del código

### 1. Importar la librería necesaria:

python

```
import pandas as pd
```

### 2. Cargar el archivo Excel:

python

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

### 3. Redondear a dos decimales la columna "Mat\_CalculoIntegral":

python

```
if 'Mat_CalculoIntegral' in df.columns:
```

```
df['Mat_CalculoIntegral'] = df['Mat_CalculoIntegral'].round(2)
```

- `df['Mat_CalculoIntegral'].round(2)`: Redondea los valores en la columna "Mat\_CalculoIntegral" a dos decimales.

#### 4. Guardar el archivo Excel modificado:

python

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

#### 5. Mostrar un mensaje indicando que el archivo se ha guardado exitosamente:

python

```
print(f'El archivo modificado ha sido guardado como "{archivo_modificado}")')
```



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 redondear a cero decimales la columna "Mat\_ProgramacionOOP"; al inicio del código agrega un comentario explicando lo que hace el programa y documenta las líneas importantes del código. Solo aplica lo que te pedí, no agregues ajustes anteriores



Aquí tienes el programa en Python que carga el archivo "calificaciones\_alumnos.xlsx" y redondea a cero decimales los valores en la columna "Mat\_ProgramacionOOP". El código incluye comentarios explicativos y solo aplica lo que solicitaste:

python

```
import pandas as pd

# Este programa carga un archivo Excel con calificaciones de alumnos,
# y redondea a cero decimales los valores en la columna "Mat_ProgramacionOOP".

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

# Redondear a cero decimales la columna "Mat_ProgramacionOOP"
if 'Mat_ProgramacionOOP' in df.columns:
    df['Mat_ProgramacionOOP'] = df['Mat_ProgramacionOOP'].round(0)
```

```
# Guardar el archivo Excel modificado
archivo_modificado = 'calificaciones_alumnos_modificado.xlsx'
df.to_excel(archivo_modificado, index=False)

print(f'El archivo modificado ha sido guardado como "{archivo_modificado}")')
```

## Explicación del código

### 1. Importar la librería necesaria:

python

```
import pandas as pd
```

### 2. Cargar el archivo Excel:

python

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

### 3. Redondear a cero decimales la columna "Mat\_ProgramacionOOP":

python

```
if 'Mat_ProgramacionOOP' in df.columns:
    df['Mat_ProgramacionOOP'] = df['Mat_ProgramacionOOP'].round(0)
```

- `df['Mat_ProgramacionOOP'].round(0)`: Redondea los valores en la columna "Mat\_ProgramacionOOP" a cero decimales (enteros).

### 4. Guardar el archivo Excel modificado:

python

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

### 5. Mostrar un mensaje indicando que el archivo se ha guardado exitosamente:

```
python
```

```
print(f'El archivo modificado ha sido guardado como "{archivo_modificado}')
```



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 redondear a cero decimales la columna "Mat\_EstructuraDatos"; al inicio del código agrega un comentario explicando lo que hace el programa y documenta las líneas importantes del código. Solo aplica lo que te pedí, no agregues ajustes anteriores



Aquí tienes el programa en Python que carga el archivo "calificaciones\_alumnos.xlsx" y redondea a cero decimales los valores en la columna "Mat\_EstructuraDatos". El código incluye comentarios explicativos y solo aplica lo que solicitaste:

```
python
```

```
import pandas as pd

# Este programa carga un archivo Excel con calificaciones de alumnos,
# y redondea a cero decimales los valores en la columna "Mat_EstructuraDatos".

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

# Redondear a cero decimales la columna "Mat_EstructuraDatos"
if 'Mat_EstructuraDatos' in df.columns:
    df['Mat_EstructuraDatos'] = df['Mat_EstructuraDatos'].round(0)

# Guardar el archivo Excel modificado
archivo_modificado = 'calificaciones_alumnos_modificado.xlsx'
df.to_excel(archivo_modificado, index=False)

print(f'El archivo modificado ha sido guardado como "{archivo_modificado}')
```

## Explicación del código

### 1. Importar la librería necesaria:

python

```
import pandas as pd
```

## 2. Cargar el archivo Excel:

python

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

## 3. Redondear a cero decimales la columna "Mat\_EstructuraDatos":

python

```
if 'Mat_EstructuraDatos' in df.columns:
    df['Mat_EstructuraDatos'] = df['Mat_EstructuraDatos'].round(0)
```

- `df['Mat_EstructuraDatos'].round(0)`: Redondea los valores en la columna "Mat\_EstructuraDatos" a cero decimales (enteros).

## 4. Guardar el archivo Excel modificado:

python

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

## 5. Mostrar un mensaje indicando que el archivo se ha guardado exitosamente:

python

```
print(f'El archivo modificado ha sido guardado como "{archivo_modificado}")
```



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 redondear a cero decimales las columnas "Mat\_CalculoIntegral", "Mat\_ProgramacionOOP", "Mat\_EstructuraDatos" y "Promedio"; al inicio del código agrega un comentario explicando lo que

hace el programa y documenta las líneas importantes del código.  
Solo aplica lo que te pedí, no agregues ajustes anteriores



Aquí tienes el programa en Python que carga el archivo "calificaciones\_alumnos.xlsx" y redondea a cero decimales las columnas "Mat\_CalculoIntegral", "Mat\_ProgramacionOOP", "Mat\_EstructuraDatos" y "Promedio". El código incluye comentarios explicativos y solo aplica lo que solicitaste:

python

```
import pandas as pd

# Este programa carga un archivo Excel con calificaciones de alumnos,
# y redondea a cero decimales las columnas "Mat_CalculoIntegral", "Mat_ProgramacionOOP",
# "Mat_EstructuraDatos" y "Promedio".

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

# Redondear a cero decimales las columnas especificadas
columnas_a_redondear = ['Mat_CalculoIntegral', 'Mat_ProgramacionOOP',
'Mat_EstructuraDatos', 'Promedio']

for columna in columnas_a_redondear:
    if columna in df.columns:
        df[columna] = df[columna].round(0)

# Guardar el archivo Excel modificado
archivo_modificado = 'calificaciones_alumnos_modificado.xlsx'
df.to_excel(archivo_modificado, index=False)

print(f'El archivo modificado ha sido guardado como "{archivo_modificado}")')
```

## Explicación del código

### 1. Importar la librería necesaria:

python

```
import pandas as pd
```

### 2. Cargar el archivo Excel:

python

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



### 3. Redondear a cero decimales las columnas especificadas:

python

```
columnas_a_redondear = ['Mat_CalculoIntegral', 'Mat_ProgramacionOOP',
                        'Mat_EstructuraDatos', 'Promedio']

for columna in columnas_a_redondear:
    if columna in df.columns:
        df[columna] = df[columna].round(0)
```

- `df[columna].round(0)`: Redondea los valores en cada columna especificada a cero decimales (enteros).

### 4. Guardar el archivo Excel modificado:

python

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

### 5. Mostrar un mensaje indicando que el archivo se ha guardado exitosamente:

python

```
print(f'El archivo modificado ha sido guardado como "{archivo_modificado}")')
```



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 convertir a letras mayúsculas las columnas "NombreAlumno", "ApellidoAlumno" y "Nombre"; al inicio del código agrega un comentario explicando lo que hace el programa y documenta las líneas importantes del código.

Solo aplica lo que te pedí, no agregues ajustes anteriores



Aquí tienes el programa en Python que carga el archivo "calificaciones\_alumnos.xlsx" y convierte a letras mayúsculas los valores en las columnas "NombreAlumno", "ApellidoAlumno" y "Nombre". El código incluye comentarios explicativos y solo aplica lo que solicitaste:

python

```
import pandas as pd

# Este programa carga un archivo Excel con calificaciones de alumnos,
# y convierte a letras mayúsculas los valores en las columnas "NombreAlumno",
# "ApellidoAlumno" y "Nombre".

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

# Convertir a letras mayúsculas las columnas especificadas
columnas_a_mayusculas = ['NombreAlumno', 'ApellidoAlumno', 'Nombre']

for columna in columnas_a_mayusculas:
    if columna in df.columns:
        df[columna] = df[columna].str.upper()

# Guardar el archivo Excel modificado
archivo_modificado = 'calificaciones_alumnos_modificado.xlsx'
df.to_excel(archivo_modificado, index=False)

print(f'El archivo modificado ha sido guardado como "{archivo_modificado}")')
```

## Explicación del código

### 1. Importar la librería necesaria:

python

```
import pandas as pd
```

### 2. Cargar el archivo Excel:

python

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

### 3. Convertir a letras mayúsculas las columnas especificadas:

python

```
columnas_a_mayusculas = ['NombreAlumno', 'ApellidoAlumno', 'Nombre']

for columna in columnas_a_mayusculas:
    if columna in df.columns:
        df[columna] = df[columna].str.upper()
```

- `df[columna].str.upper()`: Convierte todos los valores en la columna especificada a letras mayúsculas.

#### 4. Guardar el archivo Excel modificado:

python

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

#### 5. Mostrar un mensaje indicando que el archivo se ha guardado exitosamente:

python

```
print(f'El archivo modificado ha sido guardado como "{archivo_modificado}"')
```



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 convertir a letras mayúsculas las columnas "NombreAlumno", "ApellidoAlumno" y "Nombre"; a las mismas columnas quítale los acentos a los textos también. Al inicio del código agrega un comentario explicando lo que hace el programa y documenta las líneas importantes del código. Solo aplica lo que te pedí, no agregues ajustes anteriores



Aquí tienes el programa en Python que carga el archivo "calificaciones\_alumnos.xlsx", convierte a letras mayúsculas las columnas "NombreAlumno", "ApellidoAlumno" y "Nombre", y elimina los acentos de los textos en esas columnas. El código incluye comentarios explicativos y solo aplica lo que solicitaste:

python

```
import pandas as pd
import unicodedata

# Este programa carga un archivo Excel con calificaciones de alumnos,
# convierte a letras mayúsculas y elimina los acentos de los textos en
# las columnas "NombreAlumno", "ApellidoAlumno" y "Nombre".

# Función para eliminar acentos de una cadena
def quitar_acentos(texto):
    texto_normalizado = unicodedata.normalize('NFKD', texto)
    return ''.join(c for c in texto_normalizado if not unicodedata.combining(c))

# Cargar el archivo Excel
```

```

archivo_excel = 'calificaciones_alumnos.xlsx'
df = pd.read_excel(archivo_excel)

# Convertir a letras mayúsculas y quitar acentos en las columnas especificadas
columnas_a_procesar = ['NombreAlumno', 'ApellidoAlumno', 'Nombre']

for columna in columnas_a_procesar:
    if columna in df.columns:
        df[columna] = df[columna].str.upper() # Convertir a mayúsculas
        df[columna] = df[columna].apply(quitar_acentos) # Eliminar acentos

# Guardar el archivo Excel modificado
archivo_modificado = 'calificaciones_alumnos_modificado.xlsx'
df.to_excel(archivo_modificado, index=False)

print(f'El archivo modificado ha sido guardado como "{archivo_modificado}")

```

## Explicación del código

### 1. Importar las librerías necesarias:

python

```

import pandas as pd
import unicodedata

```

### 2. Definir una función para eliminar acentos:

python

```

def quitar_acentos(texto):
    texto_normalizado = unicodedata.normalize('NFKD', texto)
    return ''.join(c for c in texto_normalizado if not unicodedata.combining(c))

```

- `unicodedata.normalize('NFKD', texto)`: Normaliza el texto para separar caracteres acentuados en sus componentes base y acento.
- `''.join(c for c in texto_normalizado if not unicodedata.combining(c))`: Elimina los acentos combinados.

### 3. Cargar el archivo Excel:

python

```

archivo_excel = 'calificaciones_alumnos.xlsx'
df = pd.read_excel(archivo_excel)

```

### 4. Convertir a letras mayúsculas y quitar acentos en las columnas especificadas:

python

```
columnas_a_procesar = ['NombreAlumno', 'ApellidoAlumno', 'Nombre']

for columna in columnas_a_procesar:
    if columna in df.columns:
        df[columna] = df[columna].str.upper() # Convertir a mayúsculas
        df[columna] = df[columna].apply(quitar_acentos) # Eliminar acentos
```

## 5. Guardar el archivo Excel modificado:

python

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

## 6. Mostrar un mensaje indicando que el archivo se ha guardado exitosamente:

python

```
print(f'El archivo modificado ha sido guardado como "{archivo_modificado}")
```



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 convertir a letras mayúsculas las columnas "NombreAlumno", "ApellidoAlumno" y "Nombre"; a las mismas columnas quítale los acentos a los textos también. Redondea a cero decimales las columnas "Mat\_CalculoIntegral", "Mat\_ProgramacionOOP", "Mat\_EstructuraDatos" y "Promedio". Al inicio del código agrega un comentario explicando lo que hace el programa y documenta las líneas importantes del código. Solo aplica lo que te pedí, no agregues ajustes anteriores



Aquí tienes el programa en Python que realiza todas las tareas que solicitaste:

1. Convierte a letras mayúsculas las columnas "NombreAlumno", "ApellidoAlumno" y "Nombre".
2. Elimina los acentos de los textos en esas columnas.
3. Redondea a cero decimales las columnas "Mat\_CalculoIntegral", "Mat\_ProgramacionOOP", "Mat\_EstructuraDatos" y "Promedio".

python

```
import pandas as pd
import unicodedata

# Este programa carga un archivo Excel con calificaciones de alumnos,
# convierte a letras mayúsculas y elimina los acentos de los textos en
# las columnas "NombreAlumno", "ApellidoAlumno" y "Nombre".
# Además, redondea a cero decimales las columnas "Mat_CalculoIntegral",
# "Mat_Programacion00P", "Mat_EstructuraDatos" y "Promedio".

# Función para eliminar acentos de una cadena
def quitar_acentos(texto):
    texto_normalizado = unicodedata.normalize('NFKD', texto)
    return ''.join(c for c in texto_normalizado if not unicodedata.combining(c))

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

# Convertir a letras mayúsculas y quitar acentos en las columnas especificadas
columnas_texto = ['NombreAlumno', 'ApellidoAlumno', 'Nombre']

for columna in columnas_texto:
    if columna in df.columns:
        df[columna] = df[columna].str.upper() # Convertir a mayúsculas
        df[columna] = df[columna].apply(quitar_acentos) # Eliminar acentos

# Redondear a cero decimales las columnas de calificaciones
columnas_calificaciones = ['Mat_CalculoIntegral', 'Mat_Programacion00P',
                           'Mat_EstructuraDatos', 'Promedio']

for columna in columnas_calificaciones:
    if columna in df.columns:
        df[columna] = df[columna].round(0) # Redondear a cero decimales

# Guardar el archivo Excel modificado
archivo_modificado = 'calificaciones_alumnos_modificado.xlsx'
df.to_excel(archivo_modificado, index=False)

print(f'El archivo modificado ha sido guardado como "{archivo_modificado}")')
```

## Explicación del código

### 1. Importar las librerías necesarias:

python

```
import pandas as pd
import unicodedata
```

### 2. Definir una función para eliminar acentos:

python

```
def quitar_acentos(texto):
    texto_normalizado = unicodedata.normalize('NFKD', texto)
    return ''.join(c for c in texto_normalizado if not unicodedata.combining(c))
```

- Normaliza el texto para separar caracteres acentuados y elimina los acentos combinados.

### 3. Cargar el archivo Excel:

python

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

### 4. Convertir a letras mayúsculas y quitar acentos en las columnas especificadas:

python

```
columnas_texto = ['NombreAlumno', 'ApellidoAlumno', 'Nombre']

for columna in columnas_texto:
    if columna in df.columns:
        df[columna] = df[columna].str.upper() # Convertir a mayúsculas
        df[columna] = df[columna].apply(quitar_acentos) # Eliminar acentos
```

### 5. Redondear a cero decimales las columnas de calificaciones:

python

```
columnas_calificaciones = ['Mat_CalculoIntegral', 'Mat_ProgramacionOOP',
                           'Mat_EstructuraDatos', 'Promedio']

for columna in columnas_calificaciones:
    if columna in df.columns:
        df[columna] = df[columna].round(0) # Redondear a cero decimales
```

### 6. Guardar el archivo Excel modificado:

python

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

### 7. Mostrar un mensaje indicando que el archivo se ha guardado exitosamente:

```
python
```

```
print(f'El archivo modificado ha sido guardado como "{archivo_modificado}")
```



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 sustituir el fragmento de cadena "Luis" por Louis y sustituir el fragmento de cadena "Alberto" por "Albert" en las columnas "NombreAlumno" y "Nombre";  
Al inicio del código agrega un comentario explicando lo que hace el programa y documenta las líneas importantes del código.  
Solo aplica lo que te pedí, no agregues ajustes anteriores



Aquí tienes el programa en Python que carga el archivo "calificaciones\_alumnos.xlsx" y sustituye el fragmento de cadena "Luis" por "Louis" y "Alberto" por "Albert" en las columnas "NombreAlumno" y "Nombre". El código incluye comentarios explicativos y solo aplica lo que solicitaste:

```
python
```

```
import pandas as pd

# Este programa carga un archivo Excel con calificaciones de alumnos,
# y sustituye el fragmento de cadena "Luis" por "Louis" y "Alberto" por "Albert"
# en las columnas "NombreAlumno" y "Nombre".

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

# Función para realizar sustituciones en una cadena
def sustituir_fragmentos(texto):
    if isinstance(texto, str):
        texto = texto.replace('Luis', 'Louis')
        texto = texto.replace('Alberto', 'Albert')
    return texto

# Aplicar las sustituciones a las columnas especificadas
columnas_a_sustituir = ['NombreAlumno', 'Nombre']

for columna in columnas_a_sustituir:
    if columna in df.columns:
        df[columna] = df[columna].apply(sustituir_fragmentos)

# Guardar el archivo Excel modificado
archivo_modificado = 'calificaciones_alumnos_modificado.xlsx'
df.to_excel(archivo_modificado, index=False)
```



```
print(f'El archivo modificado ha sido guardado como "{archivo_modificado}")
```

## Explicación del código

### 1. Importar la librería necesaria:

python

```
import pandas as pd
```

### 2. Cargar el archivo Excel:

python

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

### 3. Definir una función para realizar sustituciones en una cadena:

python

```
def sustituir_fragmentos(texto):  
    if isinstance(texto, str):  
        texto = texto.replace('Luis', 'Louis')  
        texto = texto.replace('Alberto', 'Albert')  
    return texto
```

- `texto.replace('Luis', 'Louis')`: Sustituye "Luis" por "Louis".
- `texto.replace('Alberto', 'Albert')`: Sustituye "Alberto" por "Albert".

### 4. Aplicar las sustituciones a las columnas especificadas:

python

```
columnas_a_sustituir = ['NombreAlumno', 'Nombre']  
  
for columna in columnas_a_sustituir:  
    if columna in df.columns:  
        df[columna] = df[columna].apply(sustituir_fragmentos)
```

### 5. Guardar el archivo Excel modificado:

python

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

## 6. Mostrar un mensaje indicando que el archivo se ha guardado exitosamente:

python

```
print(f'El archivo modificado ha sido guardado como "{archivo_modificado}")
```



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 sustituir el fragmento de cadena "Victor" por Víctor en las columnas "NombreAlumno" y "Nombre";

Al inicio del código agrega un comentario explicando lo que hace el programa y documenta las líneas importantes del código.

Solo aplica lo que te pedí, no agregues ajustes anteriores



Aquí tienes el programa en Python que carga el archivo "calificaciones\_alumnos.xlsx" y sustituye el fragmento de cadena "Victor" por "Víctor" en las columnas "NombreAlumno" y "Nombre". El código incluye comentarios explicativos y solo aplica lo que solicitaste:

python

```
import pandas as pd

# Este programa carga un archivo Excel con calificaciones de alumnos,
# y sustituye el fragmento de cadena "Victor" por "Víctor" en las columnas
# "NombreAlumno" y "Nombre".

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

# Función para realizar la sustitución en una cadena
def sustituir_victor(texto):
    if isinstance(texto, str):
        texto = texto.replace('Victor', 'Víctor')
    return texto

# Aplicar la sustitución a las columnas especificadas
columnas_a_sustituir = ['NombreAlumno', 'Nombre']
```

```
for columna in columnas_a_sustituir:
    if columna in df.columns:
        df[columna] = df[columna].apply(sustituir_vector)

# Guardar el archivo Excel modificado
archivo_modificado = 'calificaciones_alumnos_modificado.xlsx'
df.to_excel(archivo_modificado, index=False)

print(f'El archivo modificado ha sido guardado como "{archivo_modificado}")')
```

## Explicación del código

### 1. Importar la librería necesaria:

python

```
import pandas as pd
```

### 2. Cargar el archivo Excel:

python

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

### 3. Definir una función para realizar la sustitución:

python

```
def sustituir_vector(texto):
    if isinstance(texto, str):
        texto = texto.replace('Victor', 'V́ictor')
    return texto
```

- `texto.replace('Victor', 'V́ictor')`: Sustituye "Victor" por "V́ictor".

### 4. Aplicar la sustitución a las columnas especificadas:

python

```
columnas_a_sustituir = ['NombreAlumno', 'Nombre']

for columna in columnas_a_sustituir:
    if columna in df.columns:
        df[columna] = df[columna].apply(sustituir_vector)
```

### 5. Guardar el archivo Excel modificado:

python

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

**6. Mostrar un mensaje indicando que el archivo se ha guardado exitosamente:**

python

```
print(f'El archivo modificado ha sido guardado como "{archivo_modificado}")
```

ChatGPT can make mistakes. Check important info.