¿Qué es el módulo `datetime`?

El módulo `datetime` en Python te permite manejar fechas y horas de una manera fácil y precisa. Con este módulo, puedes obtener la fecha y hora actual, calcular diferencias entre fechas, formatear fechas en diferentes estilos, y más.

Componentes principales del módulo `datetime`

El módulo `datetime` tiene varias clases importantes:

1. **`datetime`**:
   - Representa una fecha y hora combinadas.
   - Ejemplo:

```
from datetime import datetime
ahora = datetime.now()
print(ahora)  # Imprime la fecha y hora actual
```

2. **`date`**:
   - Representa solo una fecha (año, mes, día) sin la hora.
   - Ejemplo:

```
from datetime import date
hoy = date.today()
print(hoy)  # Imprime la fecha de hoy
```

3. **`time`**:
   - Representa solo la hora (horas, minutos, segundos) sin la fecha.
   - Ejemplo:

```
from datetime import time
hora_actual = time(14, 30)
print(hora_actual)  # Imprime 14:30:00
```

4. **`timedelta`**:
   - Representa una diferencia entre dos fechas o tiempos.
   - Ejemplo:

```
from datetime import timedelta
diferencia = timedelta(days=5, hours=3)
print(diferencia)  # Imprime 5 días y 3 horas
```

Funciones y métodos útiles

1. **Obtener la fecha y hora actual**:
   - `datetime.now()`: Obtiene la fecha y hora actual.

- `date.today()`: Obtiene la fecha actual sin la hora.
- Ejemplo:

```
ahora = datetime.now()
print(ahora)
```

2. **Crear fechas y horas específicas**:

- Puedes crear un objeto `datetime` o `date` especificando el año, mes, día, hora, etc.
- Ejemplo:

```
mi_cumpleaños = datetime(2024, 10, 20, 15, 30)
print(mi_cumpleaños)
```

3. **Formatear fechas y horas**:

- Puedes convertir un objeto `datetime` o `date` en una cadena con un formato específico usando el método `strftime`.
- Ejemplo:

```
formato = ahora.strftime("%d/%m/%Y %H:%M:%S")
print(formato)  # Imprime la fecha y hora en formato día/mes/año hora:minuto:segundo
```

4. **Parsear cadenas de texto a fechas**:

- Puedes convertir una cadena en un objeto `datetime` usando `strptime`.
- Ejemplo:

```
fecha_texto = "01/09/2024"
fecha_objeto = datetime.strptime(fecha_texto, "%d/%m/%Y")
print(fecha_objeto)  # Convierte la cadena en un objeto datetime
```

5. **Operaciones con `timedelta`**:

- Puedes sumar o restar días, horas, o minutos usando `timedelta`.
- Ejemplo:

```
mañana = ahora + timedelta(days=1)
print(mañana)  # Imprime la fecha y hora de mañana
```

Ejemplo práctico

Supongamos que quieres calcular cuántos días faltan para tu próximo cumpleaños:

```
from datetime import datetime

hoy = datetime.now()
mi_cumpleaños = datetime(hoy.year, 10, 20)

if mi_cumpleaños < hoy:
    mi_cumpleaños = datetime(hoy.year + 1, 10, 20)

días_faltantes = (mi_cumpleaños - hoy).days
print(f"Faltan {días_faltantes} días para mi cumpleaños.")
```

En que formato imprime la fecha y la hora dateTime?

Cuando utilizas `datetime.now()` o cualquier otra función para obtener un objeto `datetime`, el formato en el que se imprime por defecto es:

```
YYYY-MM-DD HH:MM.ssssss
```

Donde:

- `YYYY` es el año con cuatro dígitos.
- `MM` es el mes con dos dígitos (01 para enero, 02 para febrero, etc.).
- `DD` es el día con dos dígitos.
- `HH` es la hora en formato de 24 horas (00 a 23).
- `MM` son los minutos con dos dígitos.
- `SS` son los segundos con dos dígitos.
- `ssssss` son los microsegundos (opcional, puede no mostrarse si son 0).

Ejemplo:

```
from datetime import datetime

ahora = datetime.now()
print(ahora)
```

Podría imprimir algo como:

```
2024-09-03 14:45:30.123456
```

Este es el formato por defecto. Si quieres un formato diferente, puedes usar el método strftime para personalizarlo según tus necesidades.

strftime

El método `strftime` en Python se utiliza para formatear objetos de fecha y hora (`datetime`, `date`, `time`) en una cadena con un formato específico. Este método te permite representar la fecha y hora en cualquier estilo que desees, utilizando varios códigos de formato.

¿Cómo funciona `strftime`?

El método `strftime` convierte un objeto de fecha y hora en una cadena según el formato que especifiques. Los códigos de formato, que son combinaciones de caracteres especiales, indican cómo se debe estructurar la fecha y hora en la cadena de salida.

Sintaxis:

```
cadena_formateada = objeto_datetime.strftime(formato)
```

Donde:

- `objeto_datetime` es el objeto datetime, date, o time que deseas formatear.
- `formato` es una cadena que contiene códigos de formato para representar diferentes partes de la fecha y hora.

Principales códigos de formato:

- `%Y`: Año con cuatro dígitos (e.g., 2024).
- `%y`: Año con dos dígitos (e.g., 24 para 2024).
- `%m`: Mes con dos dígitos (01 a 12).
- `%B`: Nombre completo del mes (e.g., September).
- `%d`: Día del mes con dos dígitos (01 a 31).
- `%H`: Hora en formato de 24 horas (00 a 23).
- `%I`: Hora en formato de 12 horas (01 a 12).
- `%M`: Minutos con dos dígitos (00 a 59).
- `%S`: Segundos con dos dígitos (00 a 59).
- `%p`: AM o PM.
- `%A`: Nombre completo del día de la semana (e.g., Tuesday).
- `%w`: Día de la semana como un número (0 para domingo, 1 para lunes, etc.).
- `%f`: Microsegundos con seis dígitos.

Ejemplo de uso:

```
from datetime import datetime

ahora = datetime.now()
formato_personalizado = ahora.strftime("%d/%m/%Y %H:%M:%S")
print(formato_personalizado)  # Imprime algo como "03/09/2024 14:45:30" o "03/09/2024 14:45:30.123456"
```

Explicación del ejemplo:

- `%d/%m/%Y` convierte la fecha en el formato "día/mes/año" (e.g., "03/09/2024").
- `%H:%M:%S` convierte la hora en el formato "hora:minuto" (e.g., "14:45:30").

Otros ejemplos de formatos:

- "`%A, %B %d, %Y`": Lunes, septiembre 03, 2024
- "`%I:%M %p`": 02:45 PM
- "`%Y-%m-%d`": 2024-09-0