Hier ist ein ausführlicher Lernzettel zum Python-Modul **`datetime`**, der dir ganz genau erklärt, wie du damit **Datum und Zeit verwaltest**, wie du **Rechnen mit Daten machst** und wie du **an aktuelle Zeiten kommst**.

Lernzettel: Python `datetime` Modul – Datum & Zeit verstehen und rechnen

---

1. Einführung

Das Modul `datetime` ist das wichtigste Modul in Python, wenn es darum geht, mit Datum und Uhrzeit zu arbeiten. Es bietet Klassen und Funktionen, um:

- Aktuelle Zeit zu bekommen
- Datum und Uhrzeit darzustellen
- Datums- und Zeitdifferenzen zu berechnen (z. B. „5 Tage später“)
- Datumsstrings zu lesen und formatieren

---

2. Wichtige Klassen in `datetime`

- **`datetime.date`** – Nur Datum (Jahr, Monat, Tag)
- **`datetime.time`** – Nur Uhrzeit (Stunden, Minuten, Sekunden, Mikrosekunden)
- **`datetime.datetime`** – Datum und Zeit zusammen (Jahr, Monat, Tag, Stunde, Minute, Sekunde)
- **`datetime.timedelta`** – Zeitspanne / Differenz (z.B. 3 Tage, 5 Stunden)

---

3. Wie kommt man an die aktuelle Zeit?

```
from datetime import datetime

jetzt = datetime.now()  # aktuelles Datum + Zeit (lokale Zeitzone)
print(jetzt)  # z.B. 2025-07-04 15:30:12.123456

heute = datetime.today()  # ähnliches Ergebnis wie datetime.now()
print(heute)

utc_jetzt = datetime.utcnow()  # aktuelle UTC-Zeit (ohne Zeitzoneninfo)
print(utc_jetzt)
```

- `datetime.now()` gibt aktuelle lokale Zeit (mit Zeitzone, wenn eingestellt).

- `datetime.utcnow()` gibt aktuelle Zeit in UTC.

---

4. Wie erzeugt man eigene Datum/Uhrzeit-Objekte?

Man kann Datum oder Datum+Zeit auch selbst definieren:

```
from datetime import datetime, date, time

# Nur Datum (Jahr, Monat, Tag)
datum = date(2025, 7, 4)

# Nur Zeit (Stunde, Minute, Sekunde)
uhrzeit = time(15, 30, 0)

# Datum + Zeit
datum_zeit = datetime(2025, 7, 4, 15, 30, 0)
```

---

5. Formatieren und Parsen von Datumsstrings

- **`strftime()`**: datetime → formatierter String

```
jetzt = datetime.now()
print(jetzt.strftime("%Y-%m-%d %H:%M:%S"))  # 2025-07-04 15:30:00
```

- **`strptime()`**: String → datetime

```
datum_string = "2025-07-04 15:30:00"
datum_obj = datetime.strptime(datum_string, "%Y-%m-%d %H:%M:%S")
```

---

6. Rechnen mit Datum und Zeit: `timedelta`

Die Klasse `timedelta` repräsentiert eine Zeitspanne (Differenz) und ist der Schlüssel für Zeitrechnungen.

```
from datetime import timedelta

# Zeitspanne: 3 Tage, 5 Stunden, 10 Minuten
zeitspanne = timedelta(days=3, hours=5, minutes=10)

jetzt = datetime.now()

# Neues Datum berechnen
zukunft = jetzt + zeitspanne
vergangenheit = jetzt - zeitspanne

print(zukunft)
print(vergangenheit)
```

---

7. Wie kann man Differenzen berechnen?

Zwei `datetime`-Objekte voneinander subtrahieren ergibt ein `timedelta`:

```
start = datetime(2025, 7, 1, 12, 0, 0)
ende = datetime(2025, 7, 4, 15, 30, 0)

differenz = ende - start  # timedelta-Objekt

print(differenz.days)      # ganze Tage: 3
print(differenz.seconds)   # Sekundenanteil: 12600 (3,5 Stunden)
print(differenz.total_seconds())  # alle Sekunden als Float
```

---

8. Wie kann man einzelne Teile von Datum/Uhrzeit auslesen?

```
jetzt = datetime.now()

jahr = jetzt.year
monat = jetzt.month
tag = jetzt.day

stunde = jetzt.hour
minute = jetzt.minute
sekunde = jetzt.second
mikrosekunde = jetzt.microsecond
```

---

9. Zusammenfassung & Praxisanwendung

- Hol dir mit `datetime.now()` die aktuelle Zeit.
- Erzeuge eigene Datum/Uhrzeit-Objekte mit `datetime(year, month, day, ...)`.
- Rechne mit `timedelta`, z.B. `neues_datum = altes_datum + timedelta(days=5)`.
- Berechne Differenzen mit `datum2 - datum1`.
- Formatiere und parse Datum mit `strftime` und `strptime`.

---

10. Tipp für dein Projekt (Filtermanagement)

Wenn du z.B. Filter gewichten möchtest, die nur auf E-Mails in den letzten 7 Tagen reagieren, kannst du:

- Aktuelles Datum holen (z.B. `now = datetime.now()`)
- E-Mail-Datum mit `datetime.strptime()` in datetime umwandeln
- Dann mit `now - email_datum` die Differenz berechnen
- Prüfen, ob Differenz kleiner als 7 Tage ist (`timedelta(days=7)`)

Ich hoffe, dieser Lernzettel hilft dir, `datetime` sicher und effizient zu verstehen und anzuwenden! Falls du zu einzelnen Punkten Fragen hast, sag gern Bescheid.