Sehr gut. Du willst eine **sofort umsetzbare, absolut klare Entscheidungsregel**, wann du eine Klasse in Python verwenden sollst – und wann nicht. Hier kommt eine präzise, einfache und **sichere Entscheidungshilfe**, die du dir merken kannst und in deinem Projekt **direkt anwenden** kannst:

🎯 Wann du eine Klasse verwenden sollst – sofort erkennbar

> **Verwende eine Klasse, wenn du „etwas bauen willst, das Eigenschaften und Verhalten hat".**

Stell dir immer folgende Fragen:

✅ 1. Willst du mehrere Funktionen schreiben, die sich um dieselben Daten drehen?

> **→ Ja? → Klasse.**

Beispiel:

- Du arbeitest mit einem „Filter", der Name, Gewicht, Treffer hat – und Funktionen, um ihn anzuwenden.
- Du arbeitest mit einem „Benutzer", der Passwort, Token, Rechte hat – und Funktionen, um ihn zu authentifizieren.

➡️ Dann willst du eine Klasse: `class Filter:`, `class User:`

✅ 2. Willst du ein Objekt erschaffen, das man später mehrfach erzeugen, verändern oder vergleichen kann?

> **→ Ja? → Klasse.**

Beispiel:

- Jeder Filter ist ein eigenes Ding. Du willst viele Filter erzeugen.
- Jeder Benutzer hat eigene Werte und Verhalten.

➡️ Dann brauchst du Instanzen: `f1 = Filter(...)`, `u1 = User(...)`

✅ 3. Willst du Werte (Zustand) speichern, und mehrere Methoden darauf anwenden?

> **→ Ja? → Klasse.**

Zustand = Daten, die sich ändern können.

Beispiel:

- Schlüssel speichern
- Passwort intern behalten
- Anzahl Treffer mitzählen

➡️ Ohne Klasse müsstest du alle Werte manuell herumreichen – sehr fehleranfällig.

❌ Wann du KEINE Klasse brauchst – ebenfalls sofort erkennbar

Verwende KEINE Klasse, wenn du nur unabhängige Aktionen machen willst – ohne Zusammenhang.

❌ 1. Deine Funktionen haben keine gemeinsamen Daten oder Variablen?

→ Kein `self` notwendig? → **Keine Klasse.**

Beispiel:

- Eine Funktion rechnet Datumsunterschiede.
- Eine andere prüft ob ein Wort in einer Liste ist.
- Keine der Funktionen teilt Werte, kein gemeinsamer Zustand?

➡️ **→ Einfaches Modul mit Funktionen reicht.**

❌ 2. Du schreibst nur Hilfsfunktionen / Tools / einfache Rechenschritte?

→ Keine Klasse.

Beispiel:

```
def filter_email(text): ...
def is_spam(text): ...
```

➡️ Du brauchst keine Klasse, solange du keine zusammenhängenden Daten bearbeitest.

🔑 Superklare Faustregel zum Merken

Wenn du das willst …	Verwende …
Daten + passende Methoden zusammenfassen	✅ Klasse
Ein Objekt mit Eigenschaften erzeugen	✅ Klasse
Zustand speichern, verändern, lesen	✅ Klasse
Nur Berechnungen, Prüfungen, einfache Helfer schreiben	❌ Funktion
Unabhängige Tools in einem Modul bündeln	❌ Funktion

🧠 Denkweise zum Üben

Frag dich bei jedem Problem:

> **"Arbeite ich hier mit einem Objekt oder mit einer Aktion?"**

- **Objekt = Filter, User, Datei, Session, Token, Nachricht, Verbindung, …** → Klasse
- **Aktion = prüfen, speichern, berechnen, loggen, extrahieren …** → Funktion

---

Wenn du willst, kannst du mir ein Beispiel aus deinem Code zeigen, und ich sage dir sofort: Klasse oder nicht – mit Begründung. So lernst du es mit echten Fällen.