🟢 Git-Projekte mit GitHub-Integration 🟢

Projekt 1: Persönliches Tagebuch mit Git und Python

Beschreibung:

- Erstelle ein Python-Skript `tagebuch.py`, das täglich Einträge als `.txt`-Dateien speichert.
- Versioniere das Projekt lokal und lade es auf GitHub hoch.

Aufgaben:

1. **Skript erstellen:**
   - Entwickle ein Python-Skript, das per Input tägliche Einträge entgegennimmt und in `.txt`-Dateien speichert.
   - Nutze Datumsformat für Dateinamen (z.B. `2025-03-04.txt`).
2. **Funktion für Liste der Einträge:**
   - Implementiere eine Funktion, die alle `.txt`-Dateien ausgibt.
   - Versioniere diese Änderung als Commit.
3. **Funktion für Suche in Einträgen:**
   - Füge eine Suchfunktion hinzu, die Einträge nach Stichwort durchsucht.
   - Committe die Funktion auf einem separaten Branch und pushe diesen.
4. **Pull Request und Konflikt:**
   - Erstelle einen Pull Request und provoziere einen Konflikt, indem du die Suchfunktion lokal und auf GitHub änderst.
   - Löse den Konflikt und mergen die Branches.

Projekt 2: To-Do-Listen-Manager in Python

Beschreibung:

- Entwickle ein CLI-basiertes To-Do-Listen-Skript `todo.py` in Python.
- Nutze Git und GitHub für die Versionierung.

Aufgaben:

1. **Grundfunktion erstellen:**
   - Implementiere eine Funktion zum Hinzufügen von Aufgaben, die diese in einer `.txt`-Datei speichert.
   - Committe die Änderung und pushe sie zu GitHub.
2. **Funktion für abgeschlossene Aufgaben:**
   - Erstelle eine Funktion, die Aufgaben als erledigt markiert und verschiebt.
   - Nutze einen separaten Branch für diese Änderung und erstelle einen Pull Request.
3. **Erweiterung: Prioritäten setzen:**
   - Implementiere eine Funktion zum Setzen von Prioritäten (z.B. hoch, mittel, niedrig).
   - Provoziere einen Konflikt, indem du die gleiche Zeile lokal und online bearbeitest, und löse diesen.
4. **Zusatzaufgabe: JSON-Format verwenden:**
   - Ersetze die `.txt`-Datei durch eine `.json`-Datei zur Speicherung.
   - Erstelle und merge einen Branch mit dieser Änderung.

Projekt 3: Kochbuch mit Python und GitHub

Beschreibung:

- Entwickle ein Python-Skript `kochbuch.py`, das Rezepte als `.txt` speichert.
- Nutze Git für Versionierung und GitHub für die Zusammenarbeit.

Aufgaben:

1. **Rezept hinzufügen:**
    - Implementiere eine Funktion, die ein Rezept mit Zutaten und Zubereitung speichert.
    - Committe die Funktion und lade sie auf GitHub hoch.
2. **Erweiterung: Kategorien erstellen:**
    - Füge Kategorien wie `Desserts`, `Hauptspeisen` etc. hinzu.
    - Erstelle für jede Kategorie einen Branch und pushe diese zu GitHub.
3. **Konflikt provozieren und lösen:**
    - Bearbeite ein Rezept sowohl lokal als auch auf GitHub.
    - Ziehe die Änderungen und löse den Konflikt.
4. **Zusatzaufgabe: Suchfunktion:**
    - Implementiere eine Suchfunktion, die Rezepte nach Zutat filtert.
    - Nutze Branching und Pull Requests für diese Erweiterung.

Projekt 4: Taschenrechner mit Python und GitHub

Beschreibung:

- Entwickle ein Python-Skript `calculator.py` mit Grundrechenarten.
- Nutze Git für Versionierung und GitHub für Zusammenarbeit.

Aufgaben:

1. **Grundfunktionen erstellen:**
    - Implementiere Addition, Subtraktion, Multiplikation und Division.
    - Committe jede Funktion einzeln und pushe sie zu GitHub.
2. **Erweiterung: Fehlerbehandlung:**
    - Füge Try-Except-Blöcke für Fehler wie Division durch Null hinzu.
    - Nutze einen Branch und erstelle einen Pull Request für die Änderungen.
3. **Erweiterung: Erweiterte Funktionen:**
    - Implementiere Potenz- und Wurzelfunktion in separaten Branches.
    - Provoziere Konflikte und löse sie durch Merge.
4. **Zusatzaufgabe: Logging:**
    - Füge Logging für jede Berechnung hinzu und speichere die Logs in einer `.txt`-Datei.
    - Nutze GitHub Issues, um Fehler im Code zu dokumentieren.

Projekt 5: Notizen-Manager mit Python und GitHub

Beschreibung:

- Entwickle ein Python-Skript `notes.py`, das Notizen in `.txt`-Dateien speichert.
- Nutze Git für die Versionierung und GitHub für das Teilen der Notizen.

Aufgaben:

1. **Funktion zum Speichern von Notizen:**

- Entwickle eine Funktion zum Hinzufügen von Notizen und speichere diese in `.txt`-Dateien.
- Committe die Änderungen und lade sie auf GitHub hoch.

2. **Erweiterung: Kategorien für Notizen:**
   - Implementiere Kategorien (z.B. Arbeit, Persönlich) und speichere Notizen in entsprechenden Unterordnern.
   - Nutze Branches für die Kategorien und erstelle Pull Requests.
3. **Suchfunktion:**
   - Entwickle eine Suchfunktion, die Notizen nach Stichworten durchsucht.
   - Provoziere einen Konflikt, indem du dieselbe Datei lokal und auf GitHub bearbeitest.

Projekt 6: Markdown-Dokumentation mit Git und GitHub

Beschreibung:

- Erstelle ein Repository mit `.md`-Dateien zur Dokumentation eines Projekts.
- Nutze Git und GitHub, um die Versionen der Dokumentation zu verwalten.

Aufgaben:

1. **Einleitung schreiben:**
   - Erstelle eine README.md mit der Einleitung und lade sie auf GitHub hoch.
2. **Erweiterungen:**
   - Füge eine Installationsanleitung und Nutzungstipps in separaten Branches hinzu.
   - Nutze Pull Requests zum Mergen.
3. **Konflikt lösen:**
   - Bearbeite dieselbe Zeile sowohl lokal als auch auf GitHub.
   - Löse den Konflikt und mergen die Änderungen.
4. **GitHub Pages:**
   - Nutze GitHub Pages, um die Dokumentation als Webseite bereitzustellen.