TECH STARTER

1.2.5. Vim

Hausaufgaben (100 Punkte)

Lade dir für die Bearbeitung der Hausaufgabe den folgenden komprimierten tar-Ball herunter. Du erhältst ihn unter der Adresse https://marcolindner.io/assets/vim_practice_files.tar.gz. Mach am besten in deiner WSL ein `wget https://marcolindner.io/assets/vim_practice_files.tar.gz` und du erhältst die Datei. Hierzu musst du aber zuvor mit `sudo apt install wget` zunächst das wget-Tool installieren. Gib zu jeder Aufgabe einen Screenshot von der bearbeiteten Datei ab.

1. Bearbeitung einer Markdown-Datei für eine Projektbeschreibung (25 Punkte)
   Szenario: Du arbeitest an einer Projektbeschreibung in einer Markdown-Datei und musst einige Änderungen vornehmen.
   a. **Navigation und Bearbeitung:**
      - Öffne die Datei `projektbeschreibung.md` mit Vim.
        - vim projektbeschreibung.md
      - Navigiere zur dritten Überschrift (z.B. ## Zielsetzung). Verwende dazu die Suchfunktion /Zielsetzung.
        - Im Normalmodus (d.h. wir gehen mit ESC aus dem Insert-Modus raus) geben wir / und dann gefolgt vom Suchbegriff (Zielsetzung) ein.
      - Ändere das Wort "Zielsetzung" in "Projektziele" mit dem Befehl cw.
        - Wir sind mit dem Cursor auf Zielsetzung gegangen, dann haben wir cw als Befehl eingegeben, landen dann im Insert-Modus und geben das neue Wort “Projektziele” ein.
      - Füge nach der Zielsetzung einen neuen Absatz hinzu. Beende die Bearbeitung und kehre in den Normalmodus zurück.
        -
   b. **Aufräumen und Strukturieren:**
      - Suche im gesamten Dokument nach dem Wort "TODO" und ersetze es durch "ERLEDIGT" mit : Gehe dazu mit ESC raus aus dem Insert-Modus und mit : rein in den Befehlsmodus. Gib dann folgendes ein: %s/TODO/ERLEDIGT/g.
      - Lösche alle leeren Zeilen im Dokument. Suche nach leeren Zeilen mit /^$ und lösche sie mit dd.

2. Bearbeitung einer Logdatei für Fehleranalyse
   Szenario: Du analysierst eine Logdatei, um einen bestimmten Fehler zu finden und zu dokumentieren.
   a. **Suche nach Fehlern:**
      - Öffne die Datei `server.log` mit Vim.
      - Suche nach dem Begriff "ERROR" im gesamten Dokument. Navigiere durch die Treffer mit n und N.

- Kopiere den gesamten Fehlerblock (5 Zeilen über und 5 Zeilen unter dem Fehler) in eine neue Datei `error_report.txt`. Verwende dazu den Visuellen Modus (v), um die Zeilen zu markieren, und :w zum Speichern in die neue Datei.

b. Formatieren des Berichts:

- Füge eine Überschrift "Fehlerbericht" am Anfang der Datei `error_report.txt` hinzu.
- Sortiere die Fehler alphabetisch nach dem Fehlertyp. Verwende die Vim-Befehle :sort oder !sort im Normalmodus.
- Entferne alle doppelten Fehler (Zeilen), um den Bericht übersichtlicher zu machen. Verwende dazu :g/^$/d für leere Zeilen und :v/./d für doppelte Einträge.

3. Anpassung einer Konfigurationsdatei

Szenario: Du musst eine Konfigurationsdatei (`config.ini`) anpassen, um eine neue Funktion zu aktivieren.

a. Konfiguration aktualisieren:

- Öffne die Datei `config.ini` mit Vim.
- Suche nach dem Abschnitt `[FeatureX]` mit `/[FeatureX]` und navigiere zu ihm.
- Ändere den Wert der Zeile `enabled=false` auf `enabled=true`. Verwende `cw`, um die Änderung vorzunehmen.
- Füge eine neue Konfigurationseinstellung `timeout=30` unterhalb der bestehenden Zeilen in diesem Abschnitt hinzu. Verwende dazu den Normalmodus und `o`, um eine neue Zeile einzufügen.

b. Syntax prüfen:

- Suche nach allen Zeilen, die mit `#` beginnen, um sicherzustellen, dass keine wichtigen Kommentare versehentlich entfernt wurden.
- Füge am Ende der Datei einen neuen Abschnitt `[Logging]` hinzu und setze den Wert `level=info`.

4. Refactoring eines Python-Skripts

Szenario: Du refaktorisierst ein Python-Skript (`script.py`), um es lesbarer und effizienter zu machen.

a. Funktion umbenennen:

- Öffne die Datei `script.py` mit Vim.
- Suche nach der Funktion `def old_function_name()` und ändere sie in `def new_function_name()`. Verwende `cw` für das Umbenennen.
- Ersetze alle Vorkommen von `old_function_name` im gesamten Dokument durch `new_function_name` mit `:%s/old_function_name/new_function_name/g`.

b. Einrückungen anpassen:

- Navigiere zu einem Codeblock, der nicht korrekt eingerückt ist. Verwende die Suche `/{` oder `}`.
- Markiere den gesamten Block im Visuellen Modus (`V`), und rücke ihn mit `>` ein oder `<` aus.

c. Kommentarblöcke hinzufügen:

- Füge am Anfang der Datei einen Kommentarblock hinzu, der das Skript erklärt. Verwende den Einfügemodus (`i`), um den Text einzufügen.