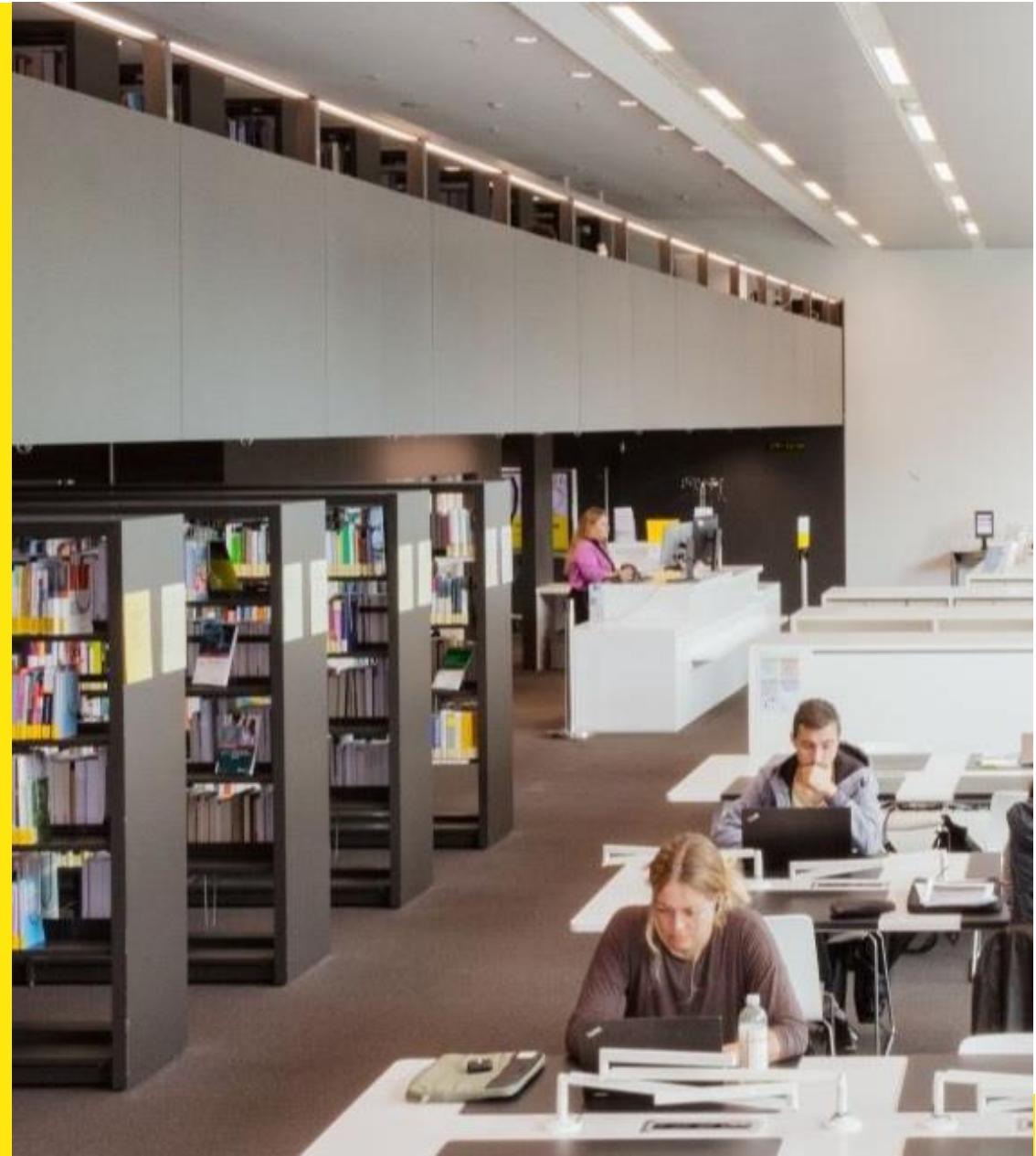


Budget-Rechner

Programmierprojekt
Grundlagen Programmierung
mit Python

WI-TZT-25
Alessio Cusintino
Jana Kräuchi
Rouven Mallach

Dezember 2025



Ablauf unserer Präsentation

- Motivation
- Ziele und -funktionen
- Projektmanagement
- Live-Demo
- Fragerunde



Motivation

- Interesse an Finanzen und Geldvorgängen
- Vereinfachte Darstellung von Budgetverwaltung
- Möglichkeit, persönliche Sparziele einfacher zu planen und einzuhalten
- Tool mit Fokus auf die Hauptaufgabe



Ziele und Funktionen

Ziele

- Einfache Verwaltung von Einnahmen und Ausgaben.
- Übersichtliche Darstellung der finanziellen Situation einer Person.
- Möglichkeit, Budgets zu setzen und zu überwachen oder Sparziele einzuhalten
- Benutzerfreundliche Bedienung für alle Altersgruppen.

Funktionen

- Einnahmen und Ausgaben erfassen
- Übersicht über Verzehr- oder Sparquote
- Bearbeitung und Löschung sämtlicher Angaben jederzeit möglich
- Festlegung von Sparzielen
- Berechnung von Sparzielen
- Anzeige von Vermögen und Budget
- Export von Daten (CSV-Datei)

Projektmanagement

- Jede:r hat ein eigenes Programm in dessen Grundstruktur gemäss des Projektbeschriebs erstellt
- grösserer Lerneffekt und bessere Einarbeitung da keiner von uns Vorkenntnisse mit Python hatte
- Flexibilität bei der Gestaltung
- Entscheidung im Team, welches Projekt schliesslich gewählt wird
- gemeinsame Ausarbeitung des gewählten Programms (Optimierung)



Projektmanagement

Verlauf:

- Harmonie im Team
- Unterstützung wo nötig
- regelmässige Check-Ins

Ergebnis:

- Zielerreichung und Lerneffekt gross
- Zusammenarbeit physisch vor Ort und Austausch zielführend via Teams



Live Demo

The screenshot shows a Mac OS X desktop environment. On the left, there is a vertical dock with icons for Finder, Mail, Safari, and other applications. In the center, a Visual Studio Code window is open, displaying a Python script named "Budget-Rechner.py". The script contains a function "bearbeite_kosten" that handles cost management. The code uses f-strings and the enumerate() function. Below the code editor, the terminal window shows the command "python3 Budget-Rechner.py" being run twice, resulting in the output "WILKOMMEN ZUM BUDGET-PLANER". At the bottom of the screen, the status bar indicates "Ln 377, Col 39" and "Python 3.9.6".

```
380 def bearbeite_kosten(benutzerdaten, kosten_art):
381     """Hilfsfunktion zum Bearbeiten von Fix- oder variablen Kosten."""
382     kosten_dict = benutzerdaten.get(kosten_art, {})
383
384     if kosten_dict == {}:
385         benutzerdaten[kosten_art] = {}
386         kosten_dict = benutzerdaten[kosten_art]
387
388     while True:
389         print(f"\n--- {kosten_art} bearbeiten ---")
390         if not kosten_dict:
391             print("Aktuell sind keine Kostenpunkte erfasst.")
392         else:
393             print("Aktuelle Kostenpunkte:")
394             for idx, (posten, wert) in enumerate(kosten_dict.items()):
395                 print(f"{idx+1}. {posten}: {format_waehrung(wert)}")
396
397             print("\nOptionen:")
398             print("A. Neuen Posten hinzufügen")
399             print("B. Bestehenden Posten ändern")
400             print("C. Bestehenden Posten löschen")
401             print("D. Zurück zur Datenanpassung")
402
403             wahl = input("Ihre Wahl (A/B/C/D): ").upper().strip()
404
405             if wahl == 'A':
406                 posten_name = input("Name des neuen Postens: ").strip()
407                 if posten_name:
408                     wert = eingabe_pruefung(f"Monatliche Kosten für {posten_name} in CHF: ")
409                     kosten_dict[posten_name] = wert
410                     print(f"Posten '{posten_name}' hinzugefügt.")
411
412             elif wahl == 'B':
413                 if not kosten_dict:
414                     print("Keine Posten zum Ändern.")
415                     continue
416
417                 print("Welche Kosten möchten Sie ändern?")
418                 kosten_name = input("Name des Postens: ").strip()
419
420                 if kosten_name in kosten_dict:
421                     kosten_wert = kosten_dict[kosten_name]
422                     print(f"Der aktuelle Wert für {kosten_name} ist {format_waehrung(kosten_wert)}.")
423
424                     neue_wert = eingabe_pruefung(f"Neuer Wert für {kosten_name} in CHF: ")
425                     kosten_dict[kosten_name] = neue_wert
426                     print(f"Der Wert für {kosten_name} wurde auf {format_waehrung(neue_wert)} geändert.")
427
428                 else:
429                     print(f"Es gibt keinen Posten mit dem Namen {kosten_name}.")
430
431             elif wahl == 'C':
432                 if not kosten_dict:
433                     print("Keine Posten zum Löschen.")
434                     continue
435
436                 kosten_name = input("Welchen Posten möchten Sie löschen? ")
437
438                 if kosten_name in kosten_dict:
439                     kosten_dict.pop(kosten_name)
440                     print(f"Der Posten '{kosten_name}' wurde gelöscht.")
441
442                 else:
443                     print(f"Es gibt keinen Posten mit dem Namen {kosten_name}.")
444
445             elif wahl == 'D':
446                 print("Zurück zur Datenanpassung...")
447
448             else:
449                 print("Ungültige Wahl. Bitte A/B/C/D eingeben.")
```

Fragen?



**Vielen Dank für die
Aufmerksamkeit.**