Modul 20 · Teamprojekt

Übersicht: alle Merk- und Infokästen mit ihren Beispielen

Aufgabe 1 – So arbeitet ihr als Team

Der Ablauf

- **1. Idee und Anforderungen:** Was soll euer Programm können? Schreibt auf, was es *mindestens* können muss und was *schön wäre*.
- **2. Entwurf:** Welche Daten speichert ihr (Listen, Texte)? Welche Funktionen braucht ihr – mit welchen Parametern und welchem Rückgabewert?
- **3. Arbeitsteilung:** Jede Person ist für bestimmte Funktionen verantwortlich. Programmiert abwechselnd und erklärt euch gegenseitig euren Code („Pair Programming“).
- **4. Testen:** Für jede Funktion schreibt ihr Testaufrufe, deren Ergebnis ihr vorher kennt.
- **5. Präsentation:** Ihr zeigt euer Programm, erklärt eure Datenstrukturen und Funktionen und berichtet, was gut lief und was schwierig war.

Das sollte euer Projekt enthalten

- mindestens eine **Liste**, die vom Programm gefüllt, durchsucht oder ausgewertet wird,
- mindestens zwei **eigene Funktionen mit Rückgabewert**,
- Verarbeitung von **Zeichenketten** (z. B. Eingaben prüfen, Texte zusammensetzen),
- **Verzweigungen** und **Schleifen**,
- **Kommentare** und sinnvolle Namen,
- **eigene Tests** für eure Funktionen (siehe Aufgabe 4).

Aufgabe 2 – Idee und Entwurf

Ideen

- **Vokabeltrainer:** Wörter und Übersetzungen in zwei Listen, Abfrage mit Punktestand, falsche Wörter werden wiederholt.
- **Notenverwaltung:** Noten einlesen, Durchschnitt, beste und schlechteste Note, Notenspiegel.
- **Highscore-Liste:** Spielergebnisse speichern, sortiert ausgeben, neue Rekorde erkennen.
- **Verschlüsselungsprogramm:** Cäsar oder eigenes Verfahren, mit Menü zum Ver- und Entschlüsseln.
- **Wetterstation (Calliope):** Temperaturen in einer Liste sammeln, Minimum, Maximum und Durchschnitt anzeigen.
- **Daten als Grafik (Turtle):** Balken- oder Kreisdiagramm aus einer Liste von Werten.

Aufgabe 4 – Testen

Eigene Tests schreiben

So wie die Funktionstests in ProgVenture könnt ihr eure Funktionen selbst prüfen: Ruft sie mit Werten auf, deren Ergebnis ihr kennt, und vergleicht.

```
print(durchschnitt([2, 3, 1]) == 2.0)
print(enthaelt(["Hund", "Katze"], "Maus") == False)
```

Gibt jede Zeile `True` aus, bestehen eure Funktionen diese Tests. Denkt auch an **Sonderfälle**: leere Listen, leere Texte, Groß- und Kleinschreibung, sehr große Zahlen.