Name: _____ Klasse: _____ Datum: _____

Listen 2: Merk dir die Folge

> **Ziel:** Du programmierst das Gedächtnisspiel „Merk dir die Folge“: Der PicoBoy zeigt eine immer längere Folge von Feldern, du spielst sie mit dem Joystick nach. Die Folge steht in einer **Liste**. (2 Stunden)



Vorbereitung: Öffne `ab16_start.py` (**Dateien → Vom Computer…**). Das Programm enthält schon fertige Funktionen:

- `zeichne_alle()` zeichnet die vier Felder.
- `leuchte(nr)` lässt Feld `nr` aufleuchten, mit Ton.
- `lies_taste()` wartet auf den Joystick und **liefert** die Feldnummer.
- `zeige_text(text)` schreibt unten einen Text.

Feldnummern: **0** oben · **1** rechts · **2** unten · **3** links. Du arbeitest nur unten im Programm, ab Zeile 45:

ab16_start.py

```
45 # ---------- Ab hier arbeitest du ----------
46 zeichne_alle()
47 folge = [0, 2, 1]
48 zeige_text("Merk dir die Folge!")
49 time.sleep_ms(800)
50 leuchte(folge[0])
51 leuchte(folge[1])
52 leuchte(folge[2])
53 zeige_text("Fertig")
```

Stunde 1: Die Folge zeigen

Aufgabe 1 ⭐ **Testen und verlängern** Teste das Programm. In welcher Reihenfolge leuchten die Farben? _____ Verlängere die Folge auf 5 Felder. Was passiert mit `leuchte(folge[5])`? _____

Aufgabe 2 ⭐ **Eine Schleife statt vieler Zeilen** Ersetze alle `leuchte`-Zeilen durch eine `for`-Schleife über die Liste `folge`. Mach daraus eine Funktion `spiele_folge(folge)` und schreibe sie zu den anderen Funktionen über Zeile 45. Teste mit einer Folge aus 6 Feldern.

Aufgabe 3 ⭐ Eine zufällige Folge Beginne mit der leeren Liste `folge = []` und hänge in einer Schleife viermal `random.randint(0, 3)` an. Gib die Liste mit `print(folge)` im Terminal aus, dann kannst du prüfen, ob die Felder richtig leuchten.

Aufgabe 4 ⭐ Erst denken, dann testen

```
1 folge = [2, 0]
2 folge.append(3)
3 folge.append(1)
4 print(len(folge))
5 print(folge[1], folge[-1])
6 for i in range(len(folge)):
7     print(i, folge[i])
```

Was gibt das Programm aus?

Stunde 2: Nachspielen und das ganze Spiel

Aufgabe 5 ⭐ Hat der Spieler recht? Die Funktion soll `True` liefern, wenn der Spieler die ganze Folge richtig nachspielt, sonst `False`. Ergänze die Bedingung und teste mit `folge = [0, 2, 1]`: Zeige „Richtig!“ oder „Leider falsch“.

```
1 def spieler_ist_richtig(folge):
2     for nr in folge:
3         taste = lies_taste()
4         leuchte(taste)
5         if                    :
6             return False
7     return True
```

Warum steht `return True` **nach** der Schleife und nicht darin?

Aufgabe 6 ⭐⭐ Das ganze Spiel Baue die Spielschleife: Die Liste beginnt leer. In jeder Runde kommt mit `append` ein zufälliges Feld dazu, unten steht „Runde 1“, „Runde 2“ …, die Folge wird gezeigt, und der Spieler spielt nach. Bei einem Fehler endet das Spiel (`break`) und zeigt, wie viele Runden geschafft wurden. Tipp: Die Rundenzahl ist `len(folge)`.

Aufgabe 7 ⭐⭐⭐ Profi-Version Wähle aus: a) Jede Runde wird schneller – gib `leuchte` einen Parameter `dauer`. b) Nach dem Spiel startet Taste A ein neues; die Ergebnisse aller Spiele stehen in einer Liste, angezeigt wird der Rekord mit `max`. c) Ein tiefer Ton bei einem Fehler.

Das habe ich gelernt:

- ☐ Ich kann eine Liste mit `append` Schritt für Schritt verlängern.
- ☐ Ich kann eine Liste mit `for` durchlaufen und mit dem Index auf Elemente zugreifen.
- ☐ Ich kann einer Funktion eine Liste übergeben.
- ☐ Ich kann eine Funktion schreiben, die `True` oder `False` liefert.

Mehr dazu im Buch: Kapitel 13 – Listen