Name: _____ Klasse: _____ Datum: _____

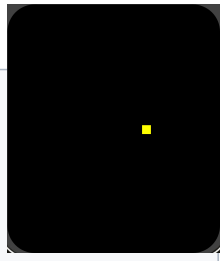
# Steuern mit dem Joystick

**Ziel:** Du steuerst eine Spielfigur mit dem Joystick über den Bildschirm, hältst sie am Rand an und verknüpfst Bedingungen mit `and`, `or` und `not`.

**So funktioniert's:** `pbc.pressed_right()` ist `True`, **solange** der Joystick nach rechts gedrückt ist. Ebenso `pressed_left()`, `pressed_up()`, `pressed_down()`, `pressed_center()` (Joystick drücken), `pressed_a()`, `pressed_b()`. **Im Emulator:** Pfeiltasten, Leertaste = MITTE, Tasten A und B. Reagiert er nicht, klicke einmal auf das Bild des PicoBoy.

### Aufgabe 1 ⭐ Die Figur bewegt sich

Öffne `ab09_start.py` (oder tippe ab) und teste mit den Pfeiltasten.



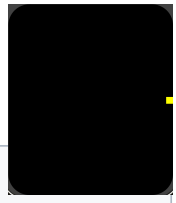
```
1 import pbc
2 import time
3 x = 116
4 y = 136
5 pbc.clear()
6 while True:
7     # Figur an der alten Stelle loeschen
8     pbc.fill_rect(x, y, 10, 10, pbc.BLACK)
9     if pbc.pressed_right():
10        x += 2
11    if pbc.pressed_left():
12        x -= 2
13    # Figur an der neuen Stelle zeichnen
14    pbc.fill_rect(x, y, 10, 10, pbc.YELLOW)
15    time.sleep_ms(20)
```

1. Ergänze zwei `if`-Abfragen, damit die Figur auch nach oben und unten fährt. Achtung: y wächst nach **unten**.
2. Mach Zeile 8 zum Kommentar (`#` davor). Was passiert jetzt?

_____

### Aufgabe 2 ⭐ Halt am Rand!

Fahre lange nach rechts: Die Figur verschwindet. Sie soll am Rand stehen bleiben. Die Figur ist 10 × 10 Pixel groß, (x, y) ist ihre linke obere Ecke, der Bildschirm ist 240 × 280 Pixel groß. Ergänze und teste alle vier Ränder.



```
if pbc.pressed_right() and x < ____:
if pbc.pressed_left() and x > ____:
if pbc.pressed_up() and y > ____:
if pbc.pressed_down() and y < ____:
```

Warum ist die rechte Grenze nicht 240? _____

### Aufgabe 3 ⭐ `and`, `or`, `not`

Wann läuft der Block? Kreuze an, welche Tasten dafür gedrückt sein können.

| Bedingung | nur A | nur B | A und B | nichts |
|---|---|---|---|---|
| `if pbc.pressed_a() and pbc.pressed_b():` | ☐ | ☐ | ☐ | ☐ |
| `if pbc.pressed_a() or pbc.pressed_b():` | ☐ | ☐ | ☐ | ☐ |
| `if not pbc.pressed_a():` | ☐ | ☐ | ☐ | ☐ |

### Aufgabe 4 ⭐⭐ Turbo

Solange B gedrückt ist, fährt die Figur schneller. Setze dafür am Anfang der Schleife `schritt = 2` und, wenn B gedrückt ist, `schritt = 6`. Ersetze dann überall `2` durch `schritt`. Am Rand musst du jetzt aufpassen: Schreibe zum Beispiel `x + schritt <= 230` statt `x < 230`. Warum?

### Aufgabe 5 ⭐⭐ Alarmfarbe

Solange A **oder** MITTE gedrückt ist, soll die Figur rot sein, sonst gelb. Speichere die Farbe in einer Variablen `farbe` und benutze sie in der Zeile, die die Figur zeichnet.

### Aufgabe 6 ⭐⭐⭐ Ins Ziel

Zeichne vor der Schleife ein grünes Ziel: `pbc.fill_rect(200, 125, 30, 30, pbc.GREEN)`. Die Figur startet links (`x = 20`). Liegt sie ganz im Ziel, erscheint „Ziel erreicht!“, und das Programm endet mit `break`. Für „ganz im Ziel“ brauchst du vier Vergleiche, die mit `and` verbunden sind. Schreibe die Bedingung auf:




**Das habe ich gelernt:**

- ☐ Ich kann mit `pbc.pressed_…()` abfragen, ob eine Taste gerade gedrückt ist.
- ☐ Ich kann eine Figur bewegen, indem ich sie lösche, ihre Position ändere und neu zeichne.
- ☐ Ich kann Bedingungen mit `and`, `or` und `not` verknüpfen.
- ☐ Ich kann eine Figur am Rand anhalten.

Mehr dazu im Buch: Kapitel 8 – Tasten und Joystick