Name: _____ Klasse: _____ Datum: _____

Projekt Weltraumspiel 2: Asteroiden aus der Liste

> **Ziel:** Asteroiden (im Programm: Meteore) fallen auf dein Raumschiff zu. Du verwaltest sie in einer Liste, erkennst Treffer und zählst die Leben.

Vorbereitung: Öffne `ab23b_start.py`. Es enthält die Lösung von AB 23a mit der Funktion `steuere`. Sind die Bilder noch hochgeladen? Wenn nicht: **Dateien → Bilder verwalten…**

Aufgabe 1 ⭐ Ein Meteor

Unter `# Aufgabe 1: Meteor anlegen`:

```
meteor = pbc.sprite("meteor.png")
meteor.set_to(random.randint(0, 220), -20)
meteor.show()
```

In der Spielschleife unter `# Aufgabe 1: Meteor bewegen` (eingerückt):

```
    meteor.set_by(0, TEMPO_METEOR)
    if meteor.y > pbc.HEIGHT:
        meteor.set_to(random.randint(0, 220), -20)
```

Warum startet der Meteor bei y = −20 und nicht bei 0? _____

Aufgabe 2 ⭐ Drei Meteore aus einer Liste

Statt einem Meteor sollen `ANZAHL = 3` Meteore fallen. Lege die Konstante oben an und schreibe die Funktion über `# Hintergrund`:

```
def nach_oben(meteor):
    meteor.set_to(random.randint(0, 220), random.randint(-150, -20))
```

Ersetze deinen Meteor aus Aufgabe 1 durch eine Liste. Ergänze die Lücken:

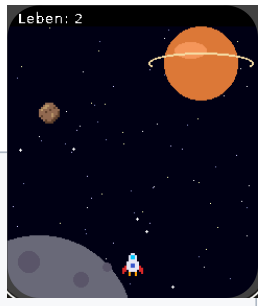
```
meteore = []
for i in range(________):
    meteor = pbc.sprite("meteor.png")
    nach_oben(meteor)
    meteor.show()
    meteore.________(meteor)
```

In der Spielschleife bewegt `for meteor in meteore:` jeden Meteor (rücke die Zeilen aus Aufgabe 1 ein und nimm `nach_oben(meteor)`). Warum fallen die Meteore versetzt und nicht alle gleichzeitig?

_____

Aufgabe 3 ⭐ Treffer!

Die Leben stehen oben auf dem Bildschirm. Weil die Sprites bei jeder Bewegung neu gezeichnet werden, kommt der Text auf das Hintergrundbild `himmel`, nicht direkt auf den Bildschirm. Schreibe diese Funktion über `# Hintergrund`:




```
def zeige_leben(leben):
    himmel.fill_rect(0, 0, 240, 20, pbc.BLACK)
    himmel.text12(f"Leben: {leben}", 10, 4, pbc.WHITE)
    pbc.show(himmel)
```

Lege vor der Spielschleife `leben = 3` an und zeige die Leben an. Prüfe in der `for`-Schleife mit `meteor.touches(schiff)`, ob ein Meteor das Schiff trifft. Dann: ein Leben weniger, ein tiefer Ton (`pbc.tone(200, 300)`), der Meteor startet neu, die Anzeige wird aktualisiert.

`touches` vergleicht die Rechtecke um die Bilder. Was bedeutet das für die Ecken?

Aufgabe 4 ⭐⭐ Game Over

Die Spielschleife soll nur laufen, solange `leben > 0` ist. Danach werden alle Sprites versteckt (`hide()`), und auf `himmel` erscheint in einem schwarzen Kasten groß „GAME OVER“. Vergiss `pbc.show(himmel)` nicht.

```

```

Aufgabe 5 ⭐⭐⭐ Jeder Meteor hat sein Tempo

Lege eine zweite Liste `tempi` an, die für jeden Meteor eine zufällige Geschwindigkeit von 4 bis 12 speichert. Tipp: Durchlaufe die Meteore mit `for i in range(len(meteore)):` und benutze `meteore[i]` und `tempi[i]`.

Das habe ich gelernt:

- ☐ Ich kann viele gleiche Sprites in einer Liste anlegen und mit `for` bewegen.
- ☐ Ich kann mit `touches` einen Treffer erkennen und Leben abziehen.
- ☐ Ich schreibe Text auf den Hintergrund, damit er nicht verschwindet.
- ☐ Ich kann ein Spiel mit `while leben > 0` enden lassen.

Mehr dazu im Buch: Kapitel 19 – Kollisionen, Kapitel 21 – Projekt: Ein Spiel Schritt für Schritt