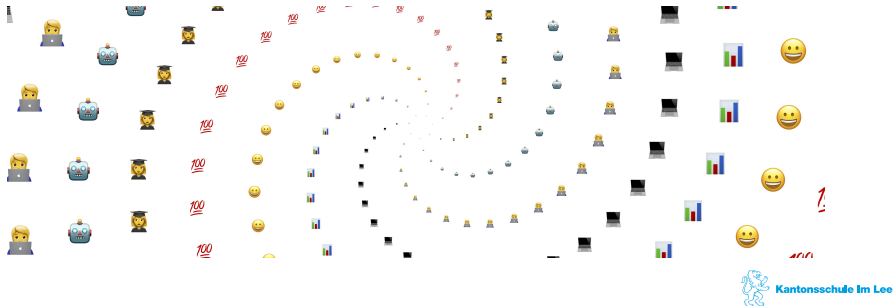


# Programmieren: Verzweigungen

Konditionale Schleifen-Abbrüche mit `while` und `break`

Cyril Wendl

Fachschaft Informatik  
Kantonsschule im Lee



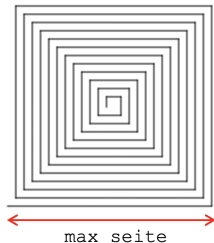
# Spirale: Abbruchkondition?

```
import turtle as t

def spirale(seite, add, max_seite):
    for _ in range(?):
        t.fd(seite)
        t.rt(90)
        side+=add

spirale(10, 10, 100)
```

Wie viele Male muss das `for _ in range(?)` ausgeführt werden?



# Spirale: Abbruchkondition?

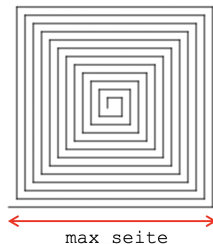
```
import turtle as t

def spirale(seite, add, max_seite):
    for _ in range(?):
        t.fd(seite)
        t.rt(90)
        side+=add
```

```
spirale(10, 10, 100)
```

Wie viele Male muss das `for _ in range(?)` ausgeführt werden?

$$\left\lceil \frac{\text{maxseite} - \text{seite}}{\text{add}} \right\rceil + 1$$



# Spirale: Abbruchkondition?

```
import turtle as t

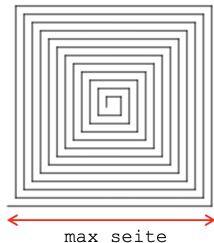
def spirale(seite, add, max_seite):
    for _ in range(?):
        t.fd(seite)
        t.rt(90)
        side+=add
```

```
spirale(10, 10, 100)
```

Wie viele Male muss das `for _ in range(?)` ausgeführt werden?

$$\left\lfloor \frac{\text{maxseite} - \text{seite}}{\text{add}} \right\rfloor + 1$$

Es geht auch einfacher ...

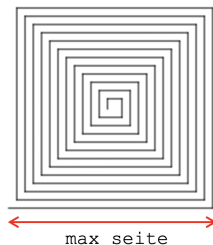


# Spirale: Flussgesteuerter Abbruch (**break**)

```
import turtle as t

def spirale(seite, add, max_seite):
    while True: ← Endlosschleife
        if seite > max_seite:
            break ← Schleifenabbruch
        t.fd(seite)
        t.rt(90)
        seite+=add

spirale(10, 10, 100)
```



# Fuss- und kopfgesteuerte Schleifen

```
import turtle as t

def spirale(seite, add, maxS):
    while True:
        if side > maxS:
            break
        t.fd(seite)
        t.rt(90)
        side+=add

spirale(10, 10, 100)
```

```
import turtle as t

def spirale(seite, add, maxS):
    while side < maxS:
        t.fd(seite)
        t.rt(90)
        side+=add

spirale(10, 10, 100)
```



# Fuss- und kopfgesteuerte Schleifen

```
import turtle as t

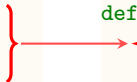
def spirale(seite, add, maxS):
    while True:
        if seite > maxS:
            break
        t.fd(seite)
        t.rt(90)
        seite+=add

spirale(10, 10, 100)
```

```
import turtle as t

def spirale(seite, add, maxS):
    while seite < maxS:
        t.fd(seite)
        t.rt(90)
        seite+=add

spirale(10, 10, 100)
```



# Wichtige Bemerkungen

- ▶ `break` bricht nicht aus `def` aus, sondern aus der (aktuellen) `while` True-Schleife!





# Wichtige Bemerkungen

- ▶ `break` bricht nicht aus `def` aus, sondern aus der (aktuellen) `while` True-Schleife!
- ▶ Ein `while` True: (unendliche Schleife) ohne `break` kann zum Absturz des Programms führen 🤪



# Wichtige Bemerkungen

- ▶ `break` bricht nicht aus `def` aus, sondern aus der (aktuellen) `while` True-Schleife!
- ▶ Ein `while` True: (unendliche Schleife) ohne `break` kann zum Absturz des Programms führen 😊
- ▶ Je nachdem kann das auch in einem `while` passieren, sofern die Bedingung nach dem `while` so geschrieben ist, dass sie immer wahr (True) ist.



# Wichtige Bemerkungen

- ▶ `break` bricht nicht aus `def` aus, sondern aus der (aktuellen) `while` True-Schleife!
- ▶ Ein `while` True: (unendliche Schleife) ohne `break` kann zum Absturz des Programms führen 😊
- ▶ Je nachdem kann das auch in einem `while` passieren, sofern die Bedingung nach dem `while` so geschrieben ist, dass sie immer wahr (True) ist.
- ▶ **Speichern Sie regelmässig Ihre Aufgaben!**



# Wichtige Bemerkungen

- ▶ `break` bricht nicht aus `def` aus, sondern aus der (aktuellen) `while` True-Schleife!
- ▶ Ein `while` True: (unendliche Schleife) ohne `break` kann zum Absturz des Programms führen 😊
- ▶ Je nachdem kann das auch in einem `while` passieren, sofern die Bedingung nach dem `while` so geschrieben ist, dass sie immer wahr (True) ist.
- ▶ **Speichern Sie regelmässig Ihre Aufgaben!**
- ▶ Falls das Programm endlos weiterläuft: Python beenden 🗑




# Aufträge

- ▶ `break`

- ▶  Aufgaben 5.24 - 5.26

- ▶  Challenge: Aufgabe 5.27

- ▶ `while` mit Bedingung:

- ▶  Aufgaben 5.28 - 5.35

