10. WHILE & FOR

DU LERNST HIER...

wie du Wiederholstrukturen mit den Schlüsselwörtern *while* und *for* verwendest. Die *while-Schleife* ist eine der wichtigsten Programmstrukturen überhaupt. Sie kann allgemein für jede Art von Wiederholungen verwendet werden und kommt in praktisch allen Programmiersprachen vor. Mit *repeat* konntest du bisher einfache Wiederholungen programmieren, ohne Variablen zu verwenden. Da du jetzt den Variablenbegriff kennst, kannst du jetzt auch die while- und for-Struktur verwenden. ▶

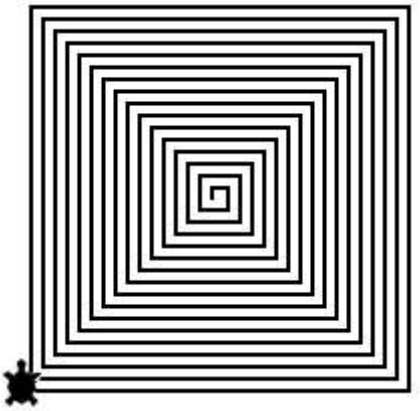
MUSTERBEISPIELE ZUR WHILE-SCHLEIFE

Eine *while-Schleife* wird mit dem Schlüsselwort **while** eingeleitet, gefolgt von einer Bedingung und einem Doppelpunkt. So lange die Bedingung erfüllt ist, werden die Befehle im nachfolgenden Programmblock wiederholt. In der Bedingung werden in der Regel die Vergleichsoperatoren >, >=, <, <=, ==, != verwendet. Die Anweisungen im *while*-Block müssen eingerückt sein.

Die Wiederholstruktur kann umgangssprachlich so formuliert werden: "*Solange die folgende Bedingung erfüllt ist, führe den nachfolgenden Programmblock aus..*"

While-Schleifen sind dann interessant, wenn die Anzahl Durchläufe nicht zum vornherein feststeht. In deinem Beispiel sollte die längste Strecke der Spirale kleiner als 200 sein.

Du beginnst du mit der Strecke a = 5. Solange a < 200 vergrösserst du nach jeder Drehung die Strecke um 2.



Programm: **[▶ Online-Editor]** **[▶ WebTigerJython]**

```
from gturtle import *

makeTurtle()
a = 5
while a < 200:
    forward(a)
    right(90)
    a = a + 2
```

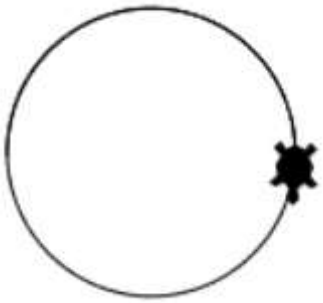
▶ In Zwischenablage kopieren

Endlose while-Schleife

Die Turtle bewegt sich "endlos" auf dem Kreis.

Mit den Schlüsselwörtern **while True**: wird eine sogenannte endlose while-Schleife eingeleitet.

Die Befehle im Schleifenblock werden so lange wiederholt, bis du das Programm mit Klick auf den roten Stop-Button beendest.



Programm: **[► Online-Editor]** **[► WebTigerJython]**

```
from gturtle import *

makeTurtle()

while True:
    forward(3)
    right(3)
```

► In Zwischenablage kopieren

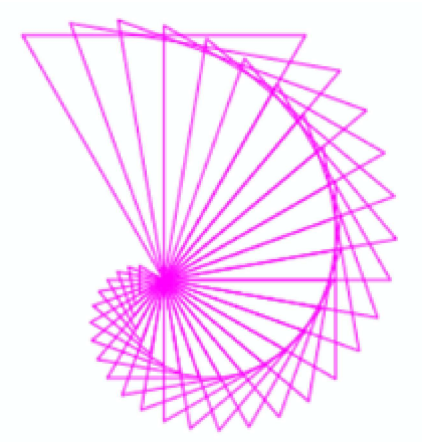
MERKE DIR...

Die Bedingung **a < 200** nennt man auch *Laufbedingung*.

Mit *while True* wird eine endlose Schleife eingeleitet, bei der die Befehle im Schleifenblock solange wiederholt werden, bis *Stop* gedrückt wird.

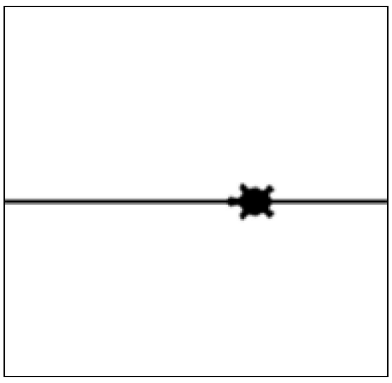
ZUM SELBST LÖSEN

1. Die Turtle zeichnet regelmässige Dreiecke, wobei das erste Dreieck die Seitenlänge s = 300 hat, das zweite 290, das dritte 280 usw. Nach jedem Dreieck folgt eine Dreuhung um 10° nach rechts. Sie beendet das Zeichnen, wenn die Seitenlänge kleiner als 10 ist.



2. Die Turtle bewegt sich in kleinen Schritten endlos im Bereich x = -250 bis 250 hin und her.

   Verwende eine Bedingung mit der Funktion *getX()*, welche die aktuelle x-Koordinate zurückgibt.



MUSTERBEISPIELE ZUR FOR-SCHLEIFE

Bei der *for*-Schleife wird der Schleifenzähler automatisch verändert.

Die Turtle bewegt sich um eine Strecke s vorwärts und dreht sich um 70° nach rechts. s wird nach jedem Durchgang um 1 erhöht. Der Schleifenzähler wird verwendet um jeweils die Streckenlänge festzulegen.

Mit **for s in range(100):** werden die Zahlen 0, 1, 2,.., 99 durchgelaufen, der Schleifenblock wird also 100 mal wiederholt.

Programm: **[► Online-Editor]** **[► WebTigerJython]**

```
from gturtle import *

makeTurtle()
hideTurtle()

for s in range(100):
    forward(s)
    right(70)
```

► In Zwischenablage kopieren

Die nächste Figur kannst du auch mit zwei *repeat*-Schleifen zeichnen. Da du in der inneren Schleife i auch für die Berechnung der Strichlänge brauchst, ist die *for*-Schleife vorteilhafter. Achte auf die korrekte Einrückung!

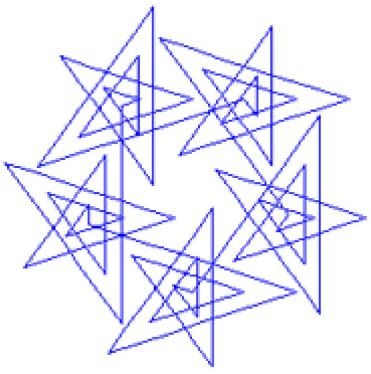
Programm: **[► Online-Editor]** **[► WebTigerJython]**

```
from gturtle import *

makeTurtle()
setPenColor("blue")

repeat 5:
    for i in range(12):
        forward(12 * i)
        left(144)
hideTurtle()
```

► In Zwischenablage kopieren

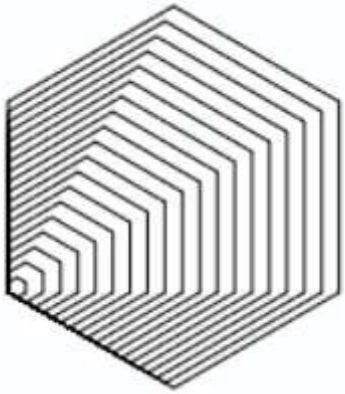


MERKE DIR...

Mit **for i in range(n):** werden die Zahlen 0, 1, 2,..(n-1) durchgelaufen. Der Parameter *n* in*range(n)* gibt die Anzahl der Wiederholungen an.

ZUM SELBST LÖSEN

3. Die nebenstehende Figur erzeugst du, indem du mit einer for-Schleife 20 Sechsecke mit immer grösseren Seitenlängen zeichnest.

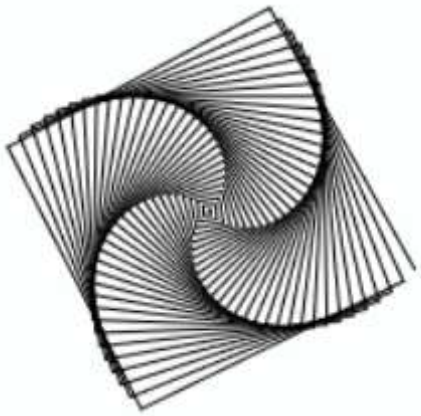


```
def hexagon(i):
    repeat 6:
        forward(i)
        right(60)
```

4. Die Turtle bewegt sich jeweils vorwärts zeichnet einen gefüllten Kreis (dot), dann rückwärts zum Ausgangspunkt und dreht um 10° nach rechts. Bei jeder Wiederholing zeichnet sie eine längere Strecke.

   Löse die Aufgabe mit einer for-Schleife und verwende für die Streckenlänge ein Vielfaches des Schleifenzählers i.

5. Erzeuge die nebenstehende Figur mit einer for-Schleife. Die Turtle beginnt mit der Streckenlänge s vorwärts und dreht dann um 89° nach rechts. Es soll von 0 bis 150 laufen.



ZUSATZSTOFF: INEINANDER GESCHACHTELTE FOR-SCHLEIFEN

Schleifen können auch ineinander geschachtelt sein. Willst du beispielsweise ein Gitter (eine Tabelle oder Matrix) mit 8 horizontalen und 8 vertikalen Zellen durchlaufen, so kannst du dies mit einem Zeilenindex *i* und einem Spaltenindex *k* tun, die beide von 0 bis 7 laufen.

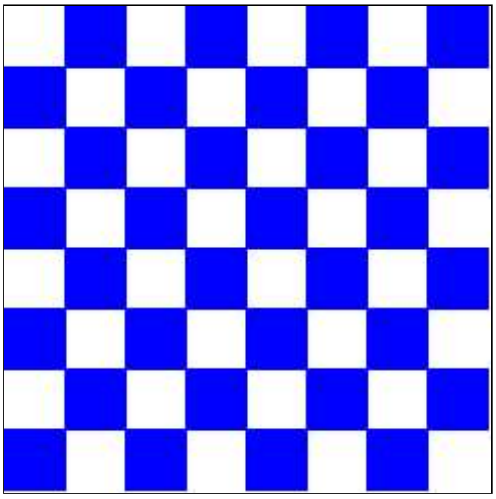
Mit

```
for i in range(8):
    for k in range(8):
        print i, k
```

durchläufst du bei festem *i* alle Spalten und nimmst dann die nächste Zeile.

	k = 0	1	2	3	4	5	6	7
i = 0	0,0	0,1	0,2	0,3	0,4	0,5	0,6	0,7
1	1,0	1,1	1,2	1,3	1,4	1,5	1,6	1,7
2	...							
3								
4								
5								
6								
7	7,0	7,1	7,2	7,3	7,4	7,5	7,6	7,7

Mit diesem Verfahren zeichnest du ein Schachbrett. Du siehst leicht ein, dass du immer dann ein gefülltes Quadrat zeichnen musst, wenn die Summe des Zeilen- und Spaltenindex eine gerade Zahl ist.



Programm: **[► WebTigerJython]** [► Online-Editor]

```
from gturtle import *

def cell(x, y):
    setPos(x, y)
    startPath()
    repeat 4:
        forward(30)
        left(90)
    fillPath()

makeTurtle()
setFillColor("blue")
hideTurtle()
for i in range(8):
    for k in range(8):
        if (i + k) % 2 == 0:
            cell(30 * i, 30 * k)
```