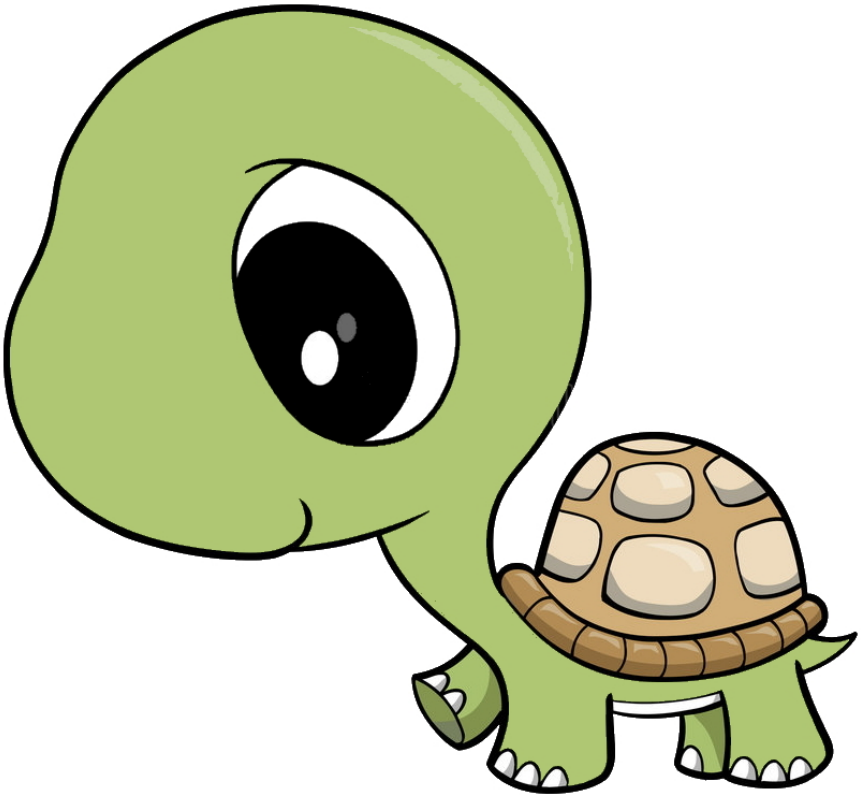
ABZ AUSBILDUNGS- UND BERATUNGSZENTRUM
FÜR INFORMATIKUNTERRICHT

Heidi Gebauer
Juraj Hromkovič
Lucia Di Caro
Ivana Kosírová
Giovanni Serafini
Jacqueline Staub
Björn Steffen

Programmieren mit LOGO

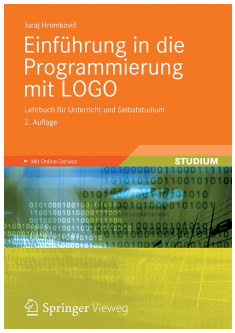


UNIVERSITÄT TRIER

ETH zürich

Programmieren mit LOGO

Dieses Skript ist eine gekürzte Version der Lektionen 1 bis 7 des Lehrbuches *Einführung in die Programmierung mit LOGO*. Das Lehrbuch enthält viele weitere Aufgaben und Erklärungen. Ausserdem ist es mit Hinweisen für die Lehrperson versehen. Das Lehrbuch umfasst insgesamt 15 Lektionen.



Juraj Hromkovič. *Einführung in die Programmierung mit LOGO: Lehrbuch für Unterricht und Selbststudium*. 3. Aufl., Springer Vieweg 2014. ISBN: 978-3-658-04832-7.

Version 3.4, 10. Februar 2023

Programmierumgebung

Die vorliegenden Unterrichtsunterlagen wurden für die Programmierumgebung XLogoOnline entwickelt. Diese ist auf der Webseite `www.xlogo.inf.ethz.ch` kostenlos verfügbar.

Nutzungsrechte

TGT

Die Turtle Group Trier der Universität Trier unterstützt Schulen und Lehrkräfte, die ihren Informatikunterricht entsprechend auf- oder ausbauen möchten, mit einem vielfältigen Angebot. Es reicht von individueller Beratung und Unterricht durch Universitäts-Professoren und das TGT-Team direkt vor Ort in den Schulen über Ausbildungs- und Weiterbildungskurse für Lehrkräfte bis zu Unterrichtsmaterialien.

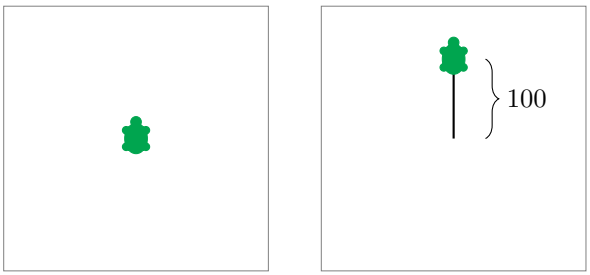
`http://xlogo.uni-trier.de/`

1 Grundbefehle

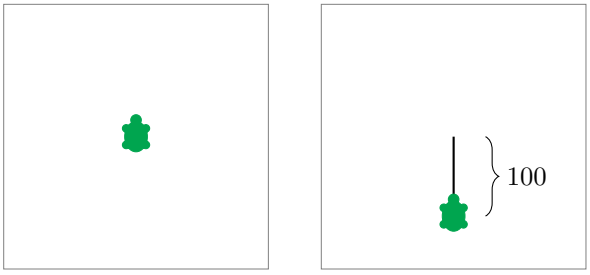
Ein **Computerbefehl** ist eine Anweisung, die der Computer versteht und ausüben kann. Der Computer kennt eigentlich nur sehr wenige Befehle und alle komplizierten Tätigkeiten, die wir vom Computer ausgeführt haben wollen, müssen wir aus den einfachen Computerbefehlen zusammensetzen. Diese Folge von Computerbefehlen nennen wir **Programm**. Programme zu schreiben ist nicht immer einfach. Es gibt Programme, die aus Millionen von Befehlen zusammengesetzt sind. Hierbei die Übersicht nicht zu verlieren, erfordert ein durchdachtes und sauberes Vorgehen, das wir in diesem Programmierkurs erlernen werden.

Gerade Linien zeichnen

Mit dem Befehl `forward 100` oder `fd 100` befiehlst du der Schildkröte 100 Schritte nach vorne zu gehen:



Mit dem Befehl `back 100` oder `bk 100` geht die Schildkröte 100 Schritte zurück:



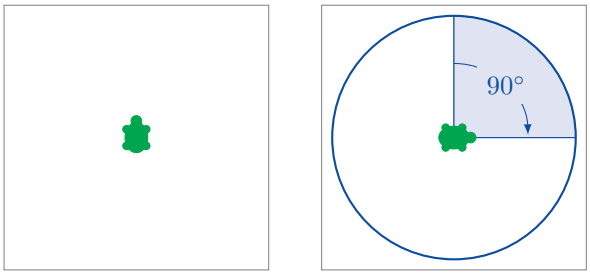
Löschen und neu beginnen

Der Befehl **`cs`** löscht alles Gezeichnete, und die Schildkröte geht in die Startposition zurück.

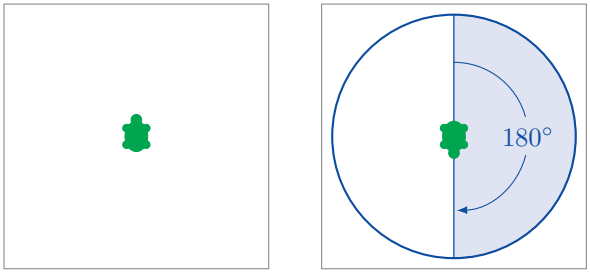
Drehen

Die Schildkröte läuft immer in die Richtung, in die sie gerade schaut.

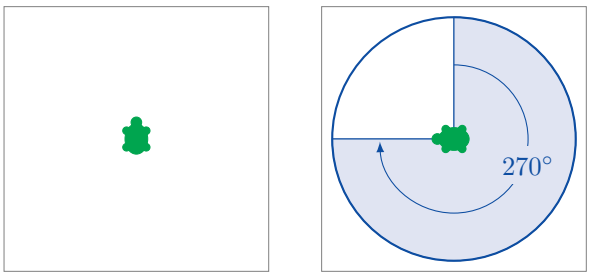
Mit dem Befehl **`right`** `90` oder **`rt`** `90` dreht sich die Schildkröte um 90° nach rechts. Dies entspricht einem Viertelkreis:



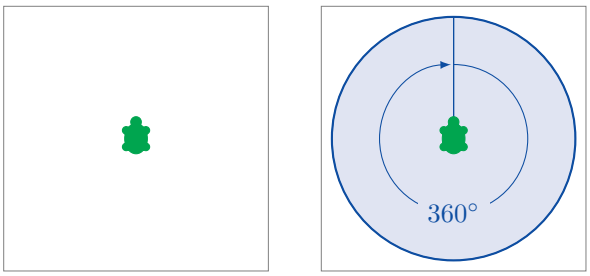
Der Befehl **`right`** `180` oder **`rt`** `180` dreht die Schildkröte um 180° nach rechts. Dies entspricht einer halben Umdrehung:



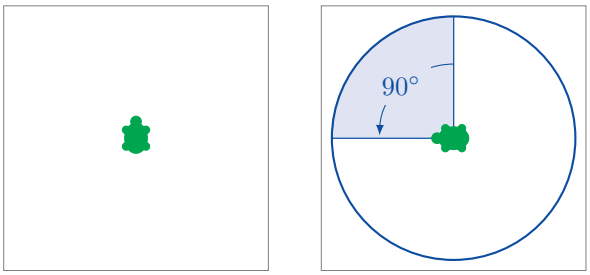
`right` `270` oder **`rt`** `270` drehen die Schildkröte um 270° nach rechts:



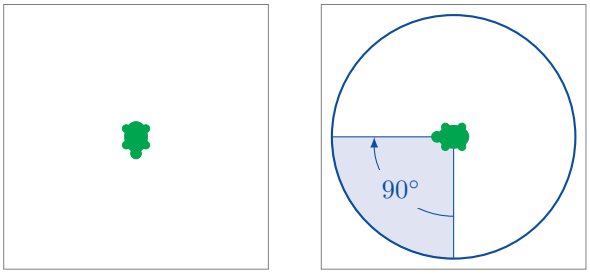
Die Befehle `right 360` und `rt 360` drehen die Schildkröte um 360° nach rechts. Dies entspricht einer vollen Umdrehung:



Mit dem Befehl `left 90` oder `lt 90` dreht sich die Schildkröte um 90° nach links:



Beachte, dass sich das Drehen nach links und rechts auf die Sicht der Schildkröte bezieht, wie das folgende Beispiel mit dem Befehl `rt 90` zeigt:



Programmieren

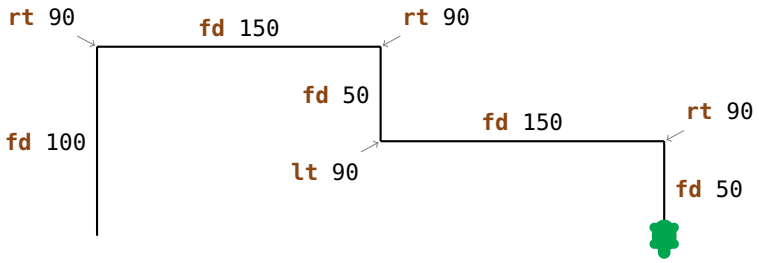
Programmieren bedeutet, eine Folge von Computerbefehlen hintereinander zu schreiben.

Aufgabe 1

Tippe das folgende Programm ab und führe es aus:

```
fd 100
rt 90
fd 150
rt 90
fd 50
lt 90
fd 150
rt 90
fd 50
```

Hast du folgendes Bild gezeichnet?



Aufgabe 2

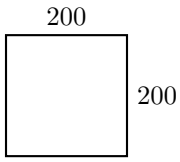
Schreibe das folgende Programm und führe es aus:

```
fd 100
rt 90
fd 200
rt 90
fd 80
rt 90
fd 100
rt 90
fd 50
```

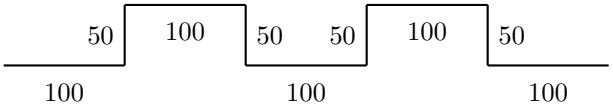
Zeichne das entstandene Bild neben das Programm oben und beschreibe (wie in Aufgabe 1) welcher Befehl was verursacht hat.

Aufgabe 3

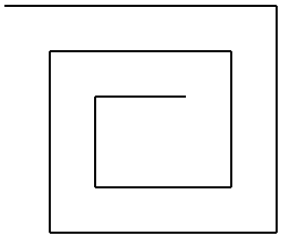
Schreibe Programme, die die folgenden Bilder zeichnen. Bei allen Bildern darfst du dir die Startposition der Schildkröte bezüglich des zu zeichnenden Bildes selbst wählen.



(a)

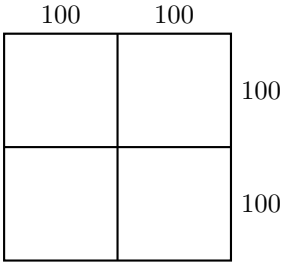


(b)



Die Größe kannst du selbst wählen.

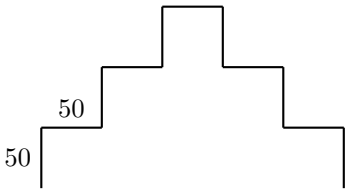
(c)



(d)

Aufgabe 4

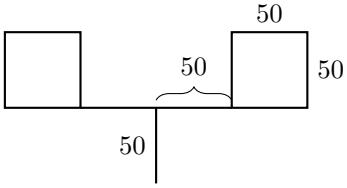
Schreibe ein Programm, das das folgende Bild zeichnet:



Schaffst du es, dein Programm so umzuschreiben, dass es nur die Befehle `fd 50` und `rt 90` verwendet?

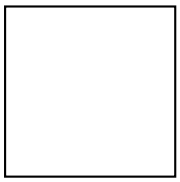
Aufgabe 5

Anna will folgendes Bild zeichnen. Kannst du ihr helfen?



2 Der Befehl `repeat`

Wenn wir ein Quadrat mit Seitenlänge 100 zeichnen wollen,



dann geht das mit dem folgenden Programm:

```
fd 100
rt 90
fd 100
rt 90
fd 100
rt 90
fd 100
rt 90
```

Wir beobachten, dass sich die beiden Befehle

```
fd 100
rt 90
```

viermal wiederholen. Wäre es nicht einfacher, dem Computer zu sagen, dass er diese zwei Befehle viermal wiederholen soll anstatt diese Befehle viermal nacheinander aufzuschreiben?

Genau dies können wir wie folgt tun:

`repeat`	`4`	`[fd 100 rt 90]`
Befehl zum Wiederholen eines Programms	Anzahl der Wieder-holungen	Folge von Befehlen, die wiederholt werden soll

Aufgabe 6

Schreibe das folgende Programm ab und führe es aus:

```
fd 75 lt 90
fd 75 lt 90
fd 75 lt 90
fd 75 lt 90
```

Was zeichnet das Programm? Kannst du den Befehl **repeat** verwenden, um das Programm kürzer zu schreiben?

Aufgabe 7

Tippe das folgende Programm ab, um zu sehen, was es zeichnet:

```
fd 50 rt 60
fd 50 rt 60
fd 50 rt 60
fd 50 rt 60
fd 50 rt 60
fd 50 rt 60
```

Schreibe es kürzer, indem du den Befehl **repeat** verwendest.

Aufgabe 8

Nutze den Befehl **repeat**, um ein Programm zum Zeichnen eines Quadrats mit der Seitenlänge 200 zu schreiben.

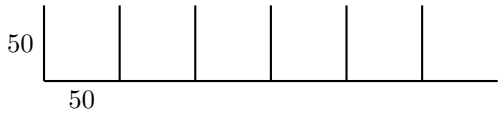
Aufgabe 9

Schreibe das folgende Programm und führe es aus:

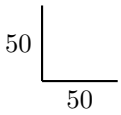
```
fd 100 rt 120
fd 100 rt 120
fd 100 rt 120
```

Was zeichnet das Programm? Kannst du den Befehl **repeat** verwenden, um das Programm kürzer zu schreiben?

Wir wollen nun das folgende Bild mit Hilfe des Befehls **repeat** zeichnen:



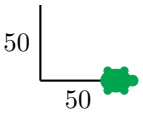
Bevor wir anfangen zu zeichnen, müssen wir uns zuerst überlegen, was das Muster ist, das sich hier wiederholt. Zum Beispiel können wir als das zu wiederholende Muster das folgende Bild nehmen:



Wenn wir in der unteren linken Ecke starten, können wir das Muster mit folgendem Programm zeichnen:

```
fd 50 bk 50 rt 90 fd 50
```

Nach der Ausführung dieses Programms steht die Schildkröte wie in der folgenden Abbildung gezeichnet und schaut nach rechts:

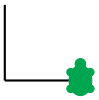


Das ist gut, weil sie schon die Position erreicht hat, an der wir das Muster nochmals zeichnen können. Sie braucht nur noch nach oben zu schauen, was wir mit dem Befehl `lt 90` erreichen können.

Wir führen das Programm aus, um zu überprüfen, ob es stimmt:

```
fd 50 bk 50 rt 90 fd 50
lt 90
```

Wir erhalten die gewünschte Situation:



Wenn wir das gleiche Programm nochmals laufen lassen, erhalten wir:

Wir sehen also, dass unsere Idee funktioniert und können unser Programm 6 Mal wiederholen:

```
repeat 6 [fd 50 bk 50 rt 90 fd 50 lt 90 ]
```

Muster — Ausrichten

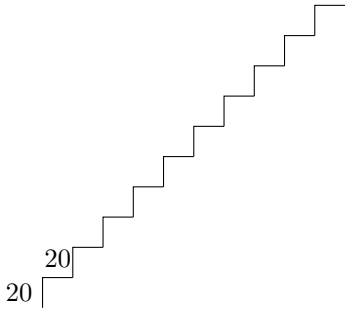
Viele Aufgaben können wir mit dieser Vorgehensweise lösen. Denke immer daran, dass du zuerst das Muster finden musst, welches sich wiederholt. Danach musst du einerseits ein Programm für das *Muster* und andererseits ein Programm für das *Ausrichten* der Schildkröte zum Zeichnen des nächsten Musters entwickeln. Dein Programm sieht dann wie folgt aus.

repeat *Anzahl* [*Muster Ausrichten*]

Aufgabe 10

Zeichnen von Treppen.

(a) Zeichne eine Treppe mit 10 Stufen der Größe 20:



- Finde zuerst das Muster, welches sich wiederholt und schreibe ein Programm dafür.
- Überlege dir ein Programm für das Ausrichten der Schildkröte, damit sie für die nächste Wiederholung des Musters richtig steht.
- Setze dann die beiden Programme geeignet zusammen, um die Aufgabe damit zu lösen.

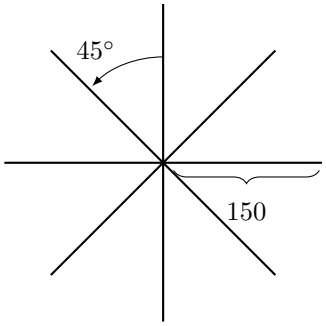
(b) Zeichne eine Treppe mit 5 Stufen der Größe 50.

(c) Zeichne eine Treppe mit 20 Stufen der Größe 10.

Aufgabe 11

Nun wollen wir Sterne zeichnen.

(a) Zeichne den folgenden Stern:

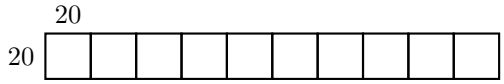


(b) Der Stern hat acht Strahlen der Länge 150. Kannst du auch einen Stern mit 16 Strahlen der Länge 100 zeichnen?

Aufgabe 12

Zeichne die folgenden Bilder:

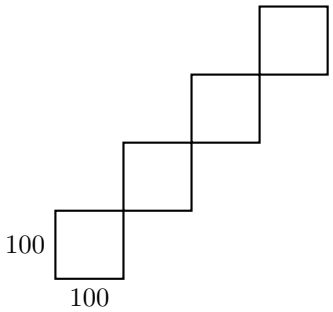
(a)



(b)

Aufgabe 13

Zeichne das folgende Bild:



Aufgabe 14

Schreibe das folgende Programm ab und führe es aus:

```
repeat 4 [fd 100 rt 90]
rt 90
repeat 4 [fd 100 rt 90]
rt 90
repeat 4 [fd 100 rt 90]
rt 90
repeat 4 [fd 100 rt 90]
rt 90
```

Was entsteht dabei? Schaffst du es, dieses Programm noch kürzer zu schreiben?

Wandermodus

Normalerweise befindet sich die Schildkröte im **Stiftmodus**. Das heisst, sie hat einen Stift in der Hand und zeichnet immer, wenn sie sich bewegt.

Im **Wandermodus** bewegt sich die Schildkröte auf dem Bildschirm ohne zu zeichnen. In den Wandermodus kommst du mit dem Befehl

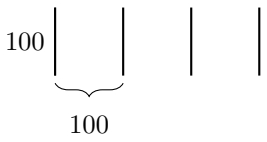
`penup` oder kurz `pu`.

Aus dem Wandermodus zurück in den Stiftmodus kommst du mit dem Befehl

`pendown` oder kurz `pd`.

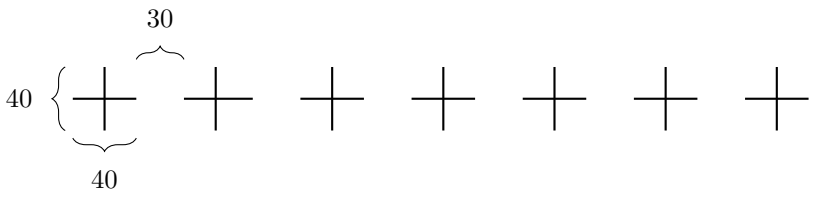
Aufgabe 15

Zeichne das folgende Bild mit einem Programm:



Aufgabe 16

Schreibe ein Programm für das folgende Bild:



3 Programme benennen und aufrufen

Jedem Programm, das wir geschrieben haben, können wir einen Namen geben. Wenn wir dann den Namen des Programms in die Befehlszeile schreiben, wird die Tätigkeit des Programms durchgeführt.

Das Programm zum Zeichnen eines Quadrats mit Seitenlänge 100 ist:

```
repeat 4 [fd 100 rt 90]
```

Wir können dieses Programm auf folgende Weise mit dem Namen `QUADRAT100` versehen:

```
to QUADRAT100
repeat 4 [fd 100 rt 90]
end
```

Wir haben also zweimal dasselbe Programm geschrieben, einmal ohne und einmal mit Namen.

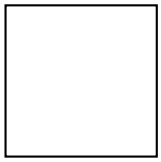
Programme mit Namen schreiben wir in den **Editor**. Solche Programme sind in diesem Heft durch eine graue Box markiert. Sobald unser Programm fertig ist, drücken wir auf den Knopf mit der Schildkröte, um den Editor wieder zu schließen.

Den Namen kann sich jeder selbst aussuchen, wir haben `QUADRAT100` gewählt, weil wir andeuten wollen, dass es um das Zeichnen eines Quadrats mit Seitenlänge 100 geht. Die einzigen Bedingungen für die Namen sind, dass sie aus Buchstaben und Zahlen und in einem Stück, ohne Abstand dazwischen, geschrieben werden.

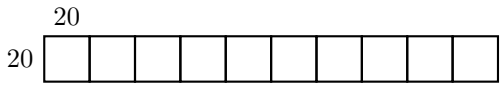
Auf dem Bildschirm wurde noch nichts gezeichnet, weil wir dem Programm nur einen Namen gegeben haben, es aber noch nicht ausgeführt haben. Wenn wir nun den Namen

```
QUADRAT100
```

in die Befehlszeile schreiben, dann wird das Programm `repeat 4 [fd 100 rt 90]` ausgeführt. Auf dem Bildschirm erscheint:



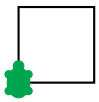
Schauen wir uns nochmals das Bild der Aufgabe 12a an.



Wir können diese Aufgabe einfacher lösen, indem wir zuerst ein Programm schreiben für das zu wiederholende Muster, also das Quadrat mit Seitenlänge 20, und ihm einen Namen geben: QUADRAT20

```
to QUADRAT20
repeat 4 [fd 20 rt 90]
end
```

Nach dem Zeichnen von QUADRAT20 steht die Schildkröte in der linken unteren Ecke des Quadrats:



Um das nächste Quadrat zu zeichnen, muss sie in die rechte untere Ecke kommen. Dies erreichen wir durch das Programm

```
rt 90 fd 20 lt 90
```

Wir werden dieses Programm ebenfalls benennen:

```
to AUSRICHTEN20
rt 90 fd 20 lt 90
end
```

Mit diesen zwei Programmen können wir ein Programm für die Aufgabe 12a wie folgt schreiben:

```
repeat 10 [QUADRAT20 AUSRICHTEN20]
```

Unser vorheriges Programm können wir auch benennen. Zum Beispiel:

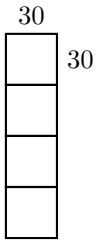
```
to REIHE10
repeat 10 [QUADRAT20 AUSRICHTEN20]
end
```

Wenn wir dies tun, nennen wir die Programme QUADRAT20 und AUSRICHTEN20 **Unterprogramme** des Programms REIHE10.

Aufgabe 17

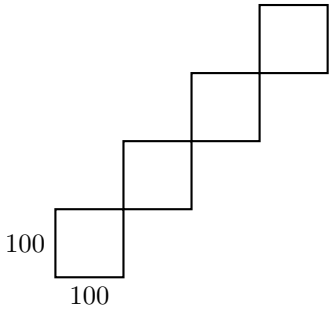
Schreibe ein Programm, welches das untenstehende Muster zeichnet und ein Programm zum Zeichnen der Quadrate mit Seitenlängen 30 verwendet. Das Programm soll wie folgt aussehen:

```
repeat 4 [QUADRAT30 AUSRICHTEN30]
```



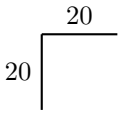
Aufgabe 18

Schreibe ein Programm `QUADRAT100` und nutze es als Unterprogramm zum Zeichnen des folgenden Bildes.

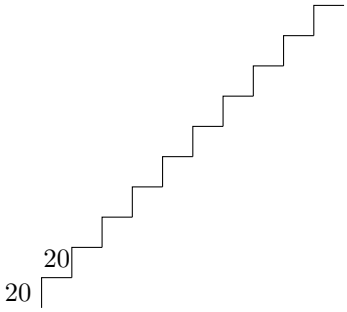


Aufgabe 19

Schreibe ein Programm zum Zeichnen einer Treppenstufe



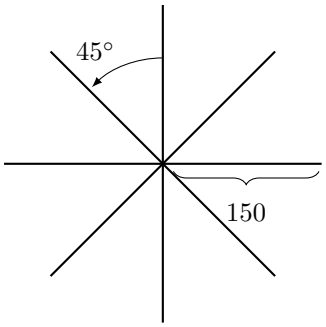
und nutze es als Unterprogramm um das folgende Bild zu zeichnen.



Aufgabe 20

Löse die untenstehende Aufgabe, indem du folgendes Unterprogramm verwendest:

```
to LINIE
fd 150 bk 150
end
```

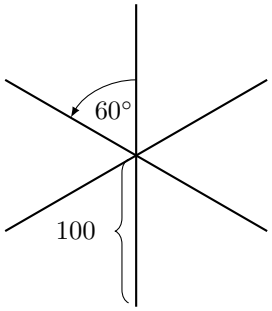


Aufgabe 21

Schreibe das folgende Programm `STRAHL` und führe es aus:

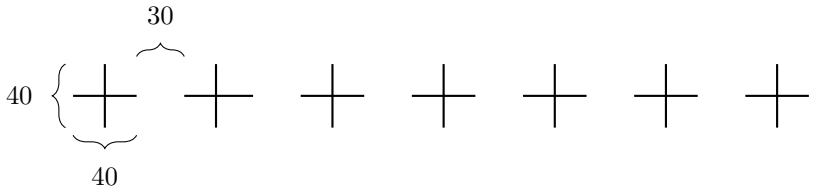
```
to STRAHL
fd 100 bk 200 fd 100
end
```

Verwende das Programm `STRAHL` als Unterprogramm für das Programm `STERN6`, welches das folgende Bild zeichnen soll:



Aufgabe 22

Zeichne das folgende Bild mit einem Unterprogramm:



Aufgabe 23

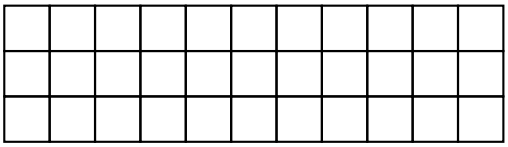
Wir haben vorher ein Programm `REIHE10` erstellt. Was macht das folgende Programm?

```
REIHE10 fd 20 lt 90 fd 200 rt 90
```

Überprüfe deine Idee auf dem Computer.

Aufgabe 24

Schreibe ein Programm, welches das folgende Bild zeichnet:



Aufgabe 25

Unterschiedlich große Quadrate zeichnen.

(a) Schreibe ein Programm, das ein Quadrat mit der Seitenlänge 50 zeichnet und benenne es `QUADRAT50`. Probiere es aus, um zu sehen, ob es das Richtige macht.

(b) Schreibe ein Programm, das ein Quadrat mit der Seitenlänge 75 zeichnet.

(c) Lasse das Programm

```
QUADRAT50
QUADRAT75
QUADRAT100
```

ausführen. Was entsteht dabei?

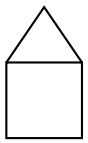
(d) Wie würdest du das Programm ändern, um noch drei weitere, größere Quadrate hinzu zu zeichnen?

Häuser bauen

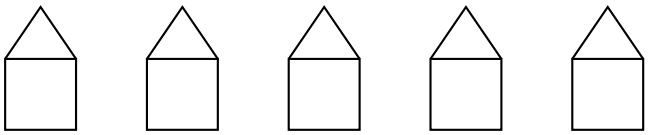
Als nächstes möchten wir einem Architekten beim Bau einer Wohnsiedlung helfen. Damit der Bau einfacher ist, möchte er alle Häuser gleich bauen. Wir geben ihm den folgenden Vorschlag:

```
to HAUS
rt 90
repeat 4 [fd 50 rt 90]
lt 60 fd 50 rt 120 fd 50 lt 150
end
```

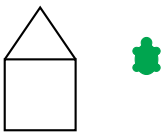
Dieses Programm zeichnet das folgende Haus:



Der Architekt hat dieses Haus aufbauen lassen und sieht jetzt, dass alles funktioniert. Deshalb verwendet er jetzt dieses Programm als Baustein, um eine erste Straße mit Häusern zu bebauen. Am Schluss soll die Straße so aussehen:



Weil er das Haus immer nach dem gleichen Muster zeichnet, kann er den Baustein `HAUS` fünfmal verwenden und muss sich nicht jedes Mal neu überlegen, wie er das Haus bauen soll. Er lässt die Schildkröte zuerst das erste Haus von links zeichnen und sagt der Schildkröte danach, dass sie an den Anfangspunkt des zweiten Hauses springen soll:



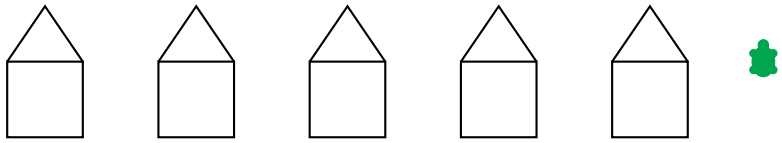
Das macht der Architekt mit dem folgenden Programm:

```
HAUS rt 90 pu fd 50 lt 90 pd
```

Nun kann die Schildkröte an diesem Ort genau das gleiche Haus nochmals zeichnen und wieder an den Anfangspunkt des nächsten Hauses springen. Das macht sie so lange, bis sie alle 5 Häuser gezeichnet hat. Wir müssen also den Programmteil oben fünfmal wiederholen und bekommen dann eine Reihe mit 5 gleichen Häusern. Das Programm dazu nennen wir `HAUSREIHE`:

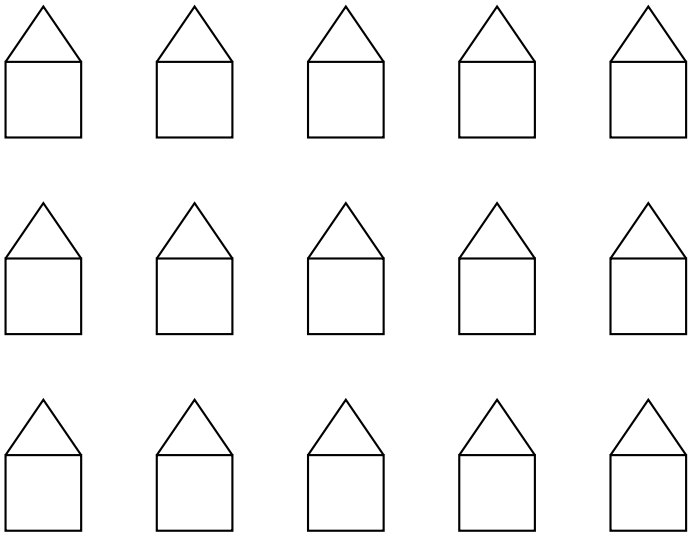
```
to HAUSREIHE
repeat 5 [HAUS rt 90 pu fd 50 lt 90 pd]
end
```

Die Schildkröte steht ganz am Schluss da, wo das nächste Haus gezeichnet werden würde:



Aufgabe 26

Nun möchten wir die Wohnsiedlung noch um ein paar Straßen erweitern. Benutze das Programm `HAUSREIHE` als Baustein, um das folgende Bild zu zeichnen:



Hinweis: Nach jeder Reihe muss die Schildkröte an die richtige Stelle springen, um die nächste Reihe zu zeichnen.

Dicke Linien und schwarze Quadrate

Aufgabe 27

Dicke Linien zeichnen mit dem Programm `FETT`.
Benenne das folgende Programm `FETT`

```
fd 100
rt 90
fd 1
rt 90
fd 100
rt 180
```

im Editor und schreibe dann

```
FETT
```

in die Befehlszeile. Was zeichnet die Schildkröte? Zeichne mit dem Bleistift auf ein Blatt, wie das Bild entstanden ist.

Aufgabe 28

Wiederhole 100 Mal das Programm `FETT` mit dem Programm

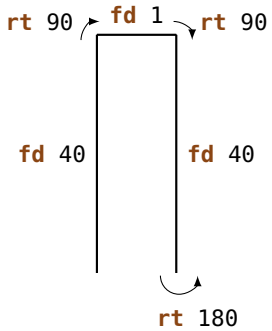
```
repeat 100 [FETT]
```

Was entsteht auf dem Bildschirm?

Aufgabe 29

In dieser Aufgabe werden wir fette Linien zeichnen. In Aufgabe 27 haben wir gesehen, dass eine fette Linie wie folgt gezeichnet werden kann:

```
to FETT40
fd 40
rt 90
fd 1
rt 90
fd 40
rt 180
end
```



Die fette Linie entsteht, indem man zwei Linien so dicht nebeneinander zeichnet, dass die beiden Linien zusammen wie eine dicke Linie aussehen.

Tippe das Programm `FETT40` ab und probiere es aus.

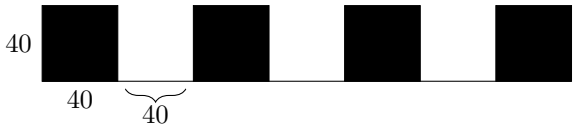
Aufgabe 30

Eine fette Linie der Länge 40 kann man als ein Rechteck mit Breite 1 und Länge 40 betrachten. Nach dem Zeichnen von `FETT40` steht die Schildkröte bei der zweiten Linie unten und schaut nach oben. Wenn das Programm `FETT40` also wiederholt wird, übermalt die Schildkröte diese zweite Linie. Wir bekommen dann also ein Rechteck mit Breite 2 und Länge 40. Mit jeder Wiederholung kommt also nur eine neue Linie hinzu. Wenn wir `FETT40` 40-mal wiederholen, entsteht das schwarze Quadrat mit der Seitenlänge 40. Probiere es aus, indem du `FETT40` 40-mal wiederholst.

Schreibe ein Programm mit dem Namen `SCHWARZ40`, das ein schwarzes Quadrat mit einer Seitenlänge von 40 zeichnet.

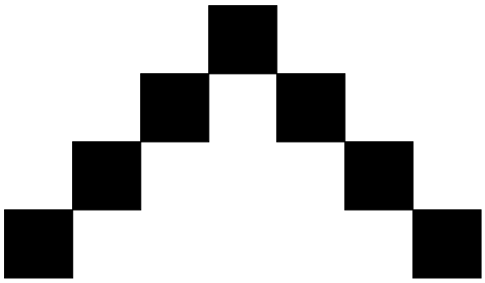
Aufgabe 31

Nutze das Programm `SCHWARZ40`, um das folgende Bild zu zeichnen:



Aufgabe 32

Verwende das Programm `SCHWARZ40`, um das folgende Bild zu zeichnen:



Aufgabe 33

Zeichne das folgende Bild:

Aufgabe 34

Schreibe ein Programm, um das folgende Bild zu zeichnen:

Aufgabe 35

Der Architekt beschließt, das Dach bei einem anderen Lieferanten zu bestellen. Er bekommt also zwei Bausteine: Einen Baustein `DACH` und einen Baustein `UNTERERTEIL`. Schreibe für den Architekten zwei Programme, die diese zwei Bausteine zeichnen, und setze sie dann in einem neuen Programm `HAUS1` zu einem Haus zusammen.

Aufgabe 36

Die Häuser in Aufgabe 26 sind ziemlich einfach gebaut. Sei kreativ und entwerfe ein neues Haus und baue damit eine neue Wohnsiedlung.

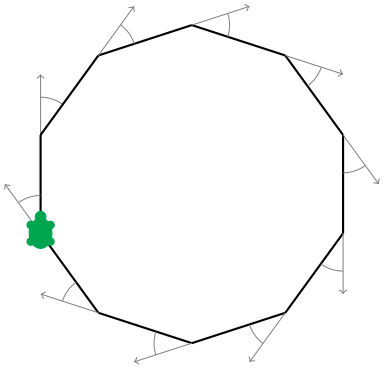
4 Regelmäßige Vielecke und Kreise

Regelmäßige Vielecke

Ein regelmäßiges k-Eck hat k Ecken und k gleich lange Seiten. Wenn du ein Vieleck, zum Beispiel ein 10-Eck, mit Bleistift zeichnen möchtest, musst du 10 Linien zeichnen und nach jeder Linie „ein bisschen“ die Richtung ändern (drehen).

Wie viel muss man drehen?

Wenn man ein Vieleck zeichnet, dreht man mehrmals unterwegs, aber am Ende steht man genau an der gleichen Stelle und schaut in genau die gleiche Richtung wie am Anfang.



Das bedeutet, dass man sich unterwegs volle 360° gedreht hat. Wenn man also ein regelmäßiges 10-Eck zeichnet, hat man sich genau zehn Mal gedreht, und zwar immer um einen gleich großen Winkel. Dieser Winkel ist:

$$\frac{360°}{10} = 36°$$

Daher muss man sich immer um 36° drehen: `rt 36`. Probieren wir das aus, indem wir folgendes Programm schreiben:

```
repeat 10 [ fd 50      rt 36   ]
```

Seitenlänge — Drehung um 36°

Aufgabe 37

Zeichne die folgenden regelmäßigen Vielecke:

(a) ein 5-Eck mit der Seitenlänge 180,

(b) ein 12-Eck mit der Seitenlänge 50,

(c) ein 4-Eck mit der Seitenlänge 200,

(d) ein 6-Eck mit der Seitenlänge 100,

(e) ein 3-Eck mit der Seitenlänge 200 und

(f) ein 18-Eck mit der Seitenlänge 20.

Wenn man ein 7-Eck zeichnen will, hat man das Problem, dass man 360 nicht ohne Rest durch 7 teilen kann. In diesem Fall lässt man das Resultat durch den Computer ausrechnen, indem man

```
360/7
```

schreibt („`/`“ bedeutet für den Computer „teile“). Der Computer findet dann das genaue Resultat. Somit kann man ein 7-Eck mit Seitenlänge 100 wie folgt zeichnen:

```
repeat 7 [fd 100 rt 360/7]
```

Probiere es aus.

Kreise zeichnen

Mit den Befehlen **`fd`** und **`rt`** kann man keine genauen Kreise zeichnen. Wie du aber sicherlich beobachtet hast, sehen Vielecke mit vielen Ecken Kreisen sehr ähnlich. Wenn wir also viele Ecken und sehr kurze Seiten nehmen, erhalten wir dadurch Kreise.

Aufgabe 38

Teste die folgenden Programme:

```
repeat 360 [fd 1 rt 1]
repeat 180 [fd 3 rt 2]
repeat 360 [fd 2 rt 1]
repeat 360 [fd 3.5 rt 1]
```

3.5 bedeutet 3 und einen halben Schritt.

Aufgabe 39

(a) Was würdest du tun, um einen ganz kleinen Kreis zu zeichnen? Schreibe ein Programm dafür.

(b) Was würdest du tun, um einen großen Kreis zu zeichnen? Schreibe ein Programm dafür.

Aufgabe 40

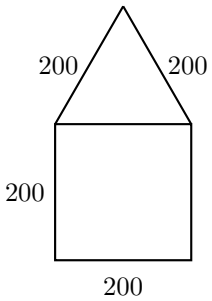
Versuche, die folgenden Halbkreise zu zeichnen. Die Größen darfst du selber bestimmen:

(a)

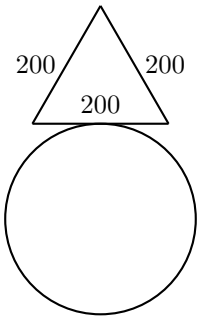
(b)

Aufgabe 41

Nutze deine neuen Erfahrungen, um die folgenden Bilder zu zeichnen. Die Größe des Kreises darfst du selber wählen:



(a)



(b)

Fantasiemuster

Zeichne ein 7-Eck mit:

```
repeat 7 [fd 100 rt 360/7]
```

Drehe dann die Schildkröte um 10° mit

```
rt 10
```

Wiederhole beides ein paar Mal, und schau dir das Bild an. Wir drehen nach jedem 7-Eck immer um 10° mit `rt 10`. Wenn wir wieder in die Ausgangsposition zurückkommen wollen, dann müssen wir diese Tätigkeit

$$\frac{360°}{10°} = 36$$

Mal wiederholen. Also schauen wir uns an, was das folgende Programm zeichnet:

```
repeat 36 [repeat 7 [fd 100 rt 360/7] rt 10]
```

Aufgabe 42

Zeichne ein regelmäßiges 12-Eck mit Seiten der Länge 70 und drehe es 18 Mal, bis du wieder an die Startposition kommst.

Hinweis: Du kannst zuerst ein Programm für ein 12-Eck mit Seitenlänge 70 schreiben und ihm zum Beispiel den Namen `ECK12` geben. Dann musst du nur noch das Programm

```
repeat 18 [ECK12 rt ... ]
```

vervollständigen.

Aufgabe 43

Denke dir eine ähnliche Aufgabe wie Aufgabe 42 aus, und schreibe ein Programm dazu.

Farben

Wenn man schon Fantasiemuster zeichnet, passen dazu auch Farben. Die Schildkröte kann nicht nur mit Schwarz, sondern mit einer beliebigen Farbe zeichnen. Jede Farbe ist durch eine Zahl bezeichnet. Eine Übersicht aller Farben findest du in der folgenden Tabelle:

0	5	9	13
1	6	10	14
2	7	11	15
3	8	12	16
4			

Mit dem Befehl

`setpencolor`	`X`
Befehl zum Ändern der Farbe	Nummer der gewünschten Farbe

wechselt die Schildkröte von der aktuellen Farbe in die Farbe mit der Nummer `X`. Wir können den Befehl durch **`setpc`** abkürzen.

Damit kann man tolle Muster zeichnen, wie zum Beispiel das Muster, das durch das folgende Programm entsteht. Zuerst benennen wir zwei Programme zum Zeichnen zweier Kreise mit unterschiedlicher Größe:

```
to KREIS3
repeat 360 [fd 3 rt 1]
end

to KREIS1
repeat 360 [fd 1 rt 1]
end
```

Jetzt nutzen wir diese Kreise, um ähnliche Muster wie die bisherigen zu entwerfen:

```
to MUST3
repeat 36 [KREIS3 rt 10]
end

to MUST1
repeat 18 [KREIS1 rt 20]
end
```

Jetzt versuchen wir es mit Farben:

```
setpc 2
MUST3 rt 2
setpc 3
MUST3 rt 2

setpc 4
MUST3 rt 2
setpc 5
MUST3 rt 2

setpc 6
MUST1 rt 2
setpc 15
MUST1 rt 2

setpc 8
MUST1 rt 2
setpc 9
MUST1 rt 2
```

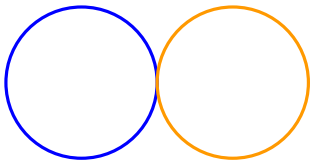
Du darfst gerne die Arbeit fortsetzen und noch mehr dazu zeichnen. Oder zeichne ein Muster nach eigener Vorstellung.

Aufgabe 44

Nutze `MUST3`, um das entsprechende Bild in Orange zu zeichnen. Verwende danach den Befehl `setpc 7`, um zur weissen Farbe zu wechseln. Was passiert jetzt, wenn du wieder `MUST3` ausführen lässt?

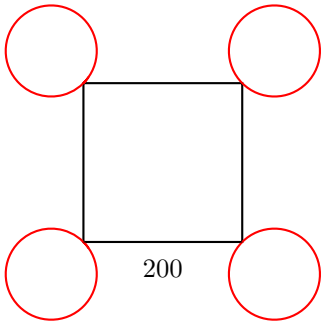
Aufgabe 45

Zeichne das folgende Bild. Die Schildkröte ist am Anfang am gemeinsamen Punkt (im Schnittpunkt) der beiden Kreise.



Aufgabe 46

Schreibe ein Programm zum Zeichnen des folgenden Bildes. Die Kreisgröße darfst du selber wählen.



5 Programme mit Parametern

In Lektion 3 haben wir gelernt, Programmen einen Namen zu geben und sie dann mit dem Namen aufzurufen, um das gewünschte Bild von dem Computer zeichnen zu lassen. In Lektion 4 haben wir gelernt, Vielecke zu zeichnen. Es ist sehr aufwendig, dass wir für jedes Vieleck mit einer neuen Anzahl von Ecken ein neues Programm schreiben müssen.

Betrachten wir zum Beispiel die folgenden drei Programme:

```
repeat 7 [fd 50 rt 360/7]
repeat 12 [fd 50 rt 360/12]
repeat 18 [fd 50 rt 360/18]
```

Die Programme sind sich sehr ähnlich und unterscheiden sich nur in den gelben Zahlen 7, 12 und 18. Diese Zahlen bestimmen die Anzahl der Ecken. Wir wollen nun ein Programm schreiben, mit dem wir alle möglichen Vielecke zeichnen können:

```
to VIELECK :ECK
repeat :ECK [fd 50 rt 360/:ECK]
end
```

Was haben wir gemacht? Überall, wo die Anzahl der Ecken im Programm steht, schreiben wir statt der Zahl einen Namen, in diesem Fall `:ECK`. Damit der Computer von vornherein weiß, dass wir die Anzahl der Ecken später frei wählen wollen, muss nach dem Namen des Programms auch `:ECK` und davor ein `:` geschrieben werden.

Wenn man jetzt den Befehl `VIELECK 12` in die Befehlszeile schreibt, setzt der Computer im Programm

```
repeat :ECK [fd 50 rt 360/:ECK]
       12                 12
```

überall, wo `:ECK` steht, die Zahl `12` ein und zeichnet so ein 12-Eck. Probiere das Ganze aus:

```
VIELECK 3
VIELECK 4
VIELECK 5
VIELECK 6
```

Wir nennen `:ECK` einen **Parameter**. Im Beispiel oben sind 3, 4, 5 und 6 **Werte des Parameters** `:ECK`. Der Computer erkennt den Parameter am `:`. Deshalb muss überall, wo ein Parameter vorkommt, ein `:` vor dem Namen des Parameters stehen.

Aufgabe 47

Die folgenden Programme zeichnen Quadrate verschiedener Seitenlängen:

```
repeat 4 [fd 100 rt 90]
repeat 4 [fd 50 rt 90]
repeat 4 [fd 200 rt 90]
```

Die gelben Zahlen 100, 50, 200 kann man als Werte eines Parameters betrachten, der die Größe des Quadrats bestimmt. Schreibe ein Programm mit dem Parameter `:GR`, das ein beliebig großes Quadrat zeichnen kann:

```
to QUADRAT :GR
...
end
```

Aufgabe 48

Die folgenden Programme zeichnen unterschiedlich große Kreise:

```
repeat 360 [fd 1 rt 1]
repeat 360 [fd 12 rt 1]
repeat 360 [fd 3 rt 1]
```

Schreibe ein Programm mit einem Parameter, mit dem man beliebig große Kreise zeichnen kann, und probiere es für die Parametergrössen 1, 2, 3, 4 und 5 aus. Den Namen des Programms und den Namen des Parameters darfst du dir selber aussuchen. Du musst nur aufpassen, dass immer der Doppelpunkt vor dem Parameter steht.

Aufgabe 49

Erinnerst du dich noch daran, wie man fette Linien zeichnen kann (Aufgabe 27)? Schreibe ein Programm mit einem Parameter, das eine fette Linie beliebiger Länge zeichnen kann.

Hinweis: Du kannst zuerst ein Programm für eine Linie der Länge 100 und ein Programm für eine Linie der Länge 50 schreiben, um zu erkennen, wo der Parameter eingesetzt werden kann.

Aufgabe 50

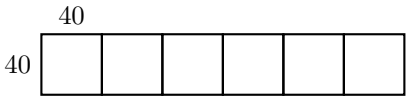
Schreibe ein Programm mit einem Parameter, das ein beliebig großes gleichseitiges Dreieck zeichnet. Zeichne dann mit diesem Programm nacheinander Dreiecke der Größen

20, 40, 60, 80, 100, 120, 140, 160 und 180.

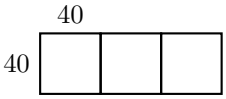
Was entsteht dabei?

Aufgabe 51

Wir wollen nun Vierecke mit der Seitenlänge 40 nebeneinander zeichnen. Schreibe ein Programm `VIERECKE` mit einem Parameter `:ANZ`. Der Parameter `:ANZ` soll die Anzahl der Vierecke bestimmen. Wenn man also `VIERECKE` 6 aufruft, soll die Schildkröte das folgende Bild zeichnen:

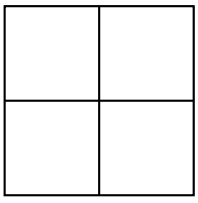


So sieht es aus, wenn man `VIERECKE` 3 aufruft:



Aufgabe 52

Schreibe ein Programm, welches das folgende Bild bestehend aus 4 Quadraten zeichnet. Die Seitenlänge der Quadrate soll durch einen Parameter bestimmt werden.

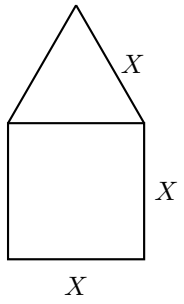


Aufgabe 53

Schreibe ein Programm mit einem Parameter, das Sechsecke beliebiger Seitenlängen zeichnet. Probiere das Programm zum Zeichnen von Sechsecken für die Seitenlängen 40, 60 und 80 aus.

Aufgabe 54

Schreibe ein Programm mit einem Parameter `:X`, das beliebig große Häuser wie in der folgenden Abbildung zeichnet.



Programm mit mehreren Parametern

Ein Programm kann mehr als einen Parameter haben. Wenn wir Vielecke zeichnen, können wir einen Parameter `:ECK` für die Anzahl der Ecken und einen Parameter `:GR` für die Seitenlänge bestimmen.

In den folgenden Programmen ist der Parameter `:ECK` mit gelb und der Parameter `:GR` mit grün markiert:

```
repeat 13 [fd 100 rt 360/13]
repeat 3 [fd 300 rt 360/3]
repeat 17 [fd 10 rt 360/17]
repeat 60 [fd 3 rt 360/60]
```

Damit können wir jetzt ein Programm für unterschiedliche Vielecke schreiben:

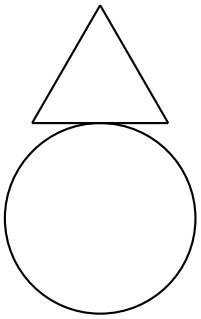
```
to VIELECKE :ECK :GR
repeat :ECK [fd :GR rt 360/:ECK]
end
```

Teste das Programm `VIELECKE` mit den folgenden Aufrufen:

```
VIELECKE 12 60
VIELECKE 12 45
VIELECKE 8 30
VIELECKE 9 30
VIELECKE 7 31
VIELECKE 11 50
```

Aufgabe 55

Schreibe ein Programm mit zwei Parametern, welches das folgende Bild zeichnen kann. Dabei soll die Kreisgröße sowie die Größe des Dreiecks frei wählbar sein.



Aufgabe 56

Das Programm

```
fd 100 rt 90 fd 200 rt 90 fd 100 rt 90 fd 200
```

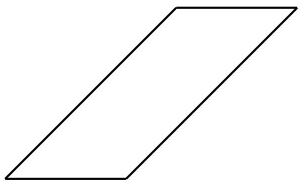
zeichnet ein Rechteck mit Breite 100 und Länge 200. Überprüfe es und schreibe ein Programm mit zwei Parametern, so dass Rechtecke mit beliebig großen Breiten und Längen gezeichnet werden können.

Aufgabe 57

Das folgende Programm

```
repeat 2 [rt 45 fd 200 rt 45 fd 100 rt 90]
```

zeichnet ein Parallelogramm:



Schreibe ein Programm mit zwei Parametern, welches solche Parallelogramme mit beliebigen Seitenlängen zeichnen kann.

Aufgabe 58

Zeichne eine Blume, indem du einen Kreis mit

```
VIELECKE 360 2
```

zeichnest, die Schildkröte dann ein bisschen drehst mit

```
rt 20
```

und anschließend wieder einen Kreis zeichnest mit

```
VIELECKE 360 2
```

und so weiter fortfährst mit `rt 20 VIELECKE 360 2 rt 20 VIELECKE 360 2 ...`

Wenn du die Blume fertig gezeichnet hast, steht die Schildkröte wieder auf der ursprünglichen Position. Die Schildkröte hat also 18 Kreise gezeichnet und sich dazwischen jeweils um 20° gedreht, somit hat sich die Schildkröte insgesamt um $18 \times 20° = 360°$ gedreht.

Zusammengefasst ergibt dies das folgende Programm:

```
repeat 18 [VIELECKE 360 2 rt 20]
```

Probiere es aus.

(a) Du kannst aber auch Blumen mit 10 Blättern (Kreisen) oder mit 20 Blättern (Kreisen) zeichnen. Wie würdest du das machen? Schreibe ein Programm dazu und probiere es aus.

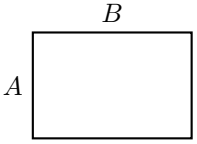
(b) Kannst du ein Programm mit einem Parameter schreiben, mit dem man Blumen mit beliebig vielen Blättern (Kreisen) zeichnen könnte?

(c) Schaffst du es, ein Programm zu schreiben, mit dem du folgende Parameter frei wählen kannst:

- die Anzahl der Blätter (Kreise) und
- die Größe der Kreise?

Aufgabe 59

Schreibe ein Programm zum Zeichnen beliebiger Rechtecke in beliebiger Farbe:



Dies bedeutet, dass die Seitenlängen A und B sowie die Farbe frei wählbar sind.

6 Blumen zeichnen und Parameter an Unterprogramme übergeben

In dieser Lektion lernen wir, Blumen zu zeichnen. Wir werden ihre Form und ihre Farbe mit der Hilfe von Parametern so wählen, dass unsere Schildkröte schöne, farbige, fantasievolle Muster zeichnen kann.

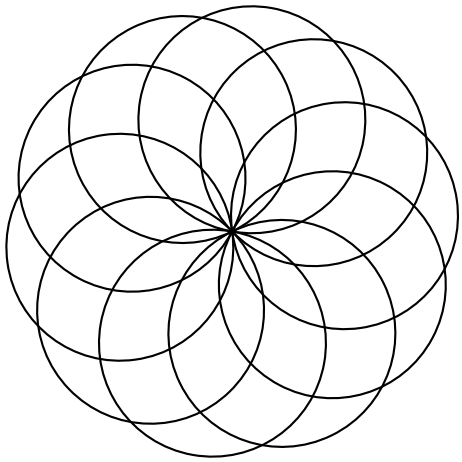
Betrachten wir nun das Programm:

```
to KREISE :GR
repeat 360[fd :GR rt 1]
end
```

Dieses Programm haben wir schon im Editor. Jetzt können wir eine Blume mit 10 Blättern mit dem Programm

```
repeat 10 [KREISE 1 rt 36]
```

zeichnen.



Aufgabe 60

Jemand will eine Blume mit 24 Blättern zeichnen. Wie müssen wir das Programm oben ändern?

Aufgabe 61

Zeichne eine Blume mit 12 Blättern und doppelt so großen Blättern wie vorher.

Jetzt wollen wir ein Programm für Blumen im Editor schreiben, bei dem die Größe der Blätter wählbar ist. Das bedeutet, wir wollen das Unterprogramm `KREISE :GR` verwenden und somit für `:GR` freie Wahl haben. Das geht nur, wenn das Programm für die Blume auch den Parameter für die Wahl der Größe der Blätter enthält.

Schreibe in den Editor

```
to BLUME :GR
repeat 10 [KREISE :GR rt 36]
end
```

Rufe `BLUME 1`, `BLUME 2` und `BLUME 3` auf und schaue dir die Zeichnung an. Was ist passiert?

Wenn wir `BLUME 1` aufgerufen haben, wurde 1 in `:GR` als Wert gesetzt. Somit wird das Unterprogramm `KREISE :GR` als `KREISE 1` aufgerufen.

Aufgabe 62

Beschreibe, was beim Aufruf `BLUME 2` passiert.

Aufgabe 63

Überlege dir, was das folgende Programm tut und dann überprüfe es.

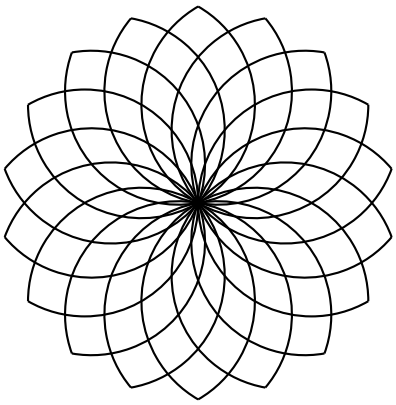
```
to BLUMEN :GR1 :GR2
setpc 3 BLUME :GR1
setpc 4 BLUME :GR2
end
```

Aufgabe 64

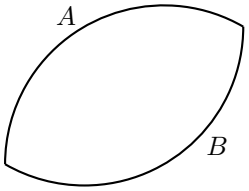
Wir wollen das Programm `BLUME` zu `BLUME1` so weiterentwickeln, dass nicht nur die Blattgröße, sondern auch die Anzahl der Blätter frei wahlbar ist. Wie machst du dies?

Eine Blume mit spitzen Blättern

Möchtest du lernen, eine Blume mit spitzen Blättern zu zeichnen? Wie gefällt dir zum Beispiel diese Blume?



Um eine solche Blume zu zeichnen, müssen wir uns zuerst überlegen, wie wir ein einzelnes Blatt zeichnen können. Ein Blatt



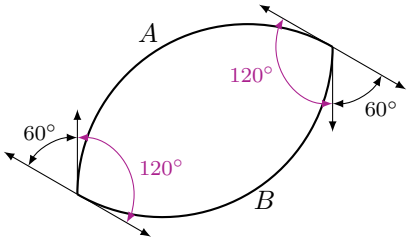
können wir als zwei zusammengeklebte Teilkreise A und B ansehen. Einen Teilkreis können wir zum Beispiel mit folgendem Programm zeichnen:

```
repeat 120 [fd 2 rt 1]
```

Probiere es aus.

Wir sehen, dass dieses Programm sehr ähnlich zum Programm für Kreise ist. Anstatt 360 kleine Bewegungen mit jeweils 1° Drehung machen wir nur 120 kleine Bewegungen `[fd 2 rt 1]` und zeichnen dadurch nur ein Drittel des Kreises (120°).

Jetzt ist die Frage, um wieviel wir die Schildkröte drehen müssen, bevor wir den Teilkreis B für die untere Seite des Blattes zeichnen. Schauen wir uns das im folgenden Bild an:



Wenn wir am Ende die ursprüngliche Position erreichen wollen, müssen wir die Schildkröte insgesamt wie immer um 360° drehen. Im Teil A drehen wir sie um 120° und im Teil B ebenfalls um 120°. Dann bleiben also noch

$$360° - 120° - 120° = 120°$$

übrig, die wir gleichmässig auf die zwei Drehungen an den Spitzen des Blattes verteilen müssen:

$$\frac{120°}{2} = 60°.$$

Damit erhalten wir folgendes Programm:

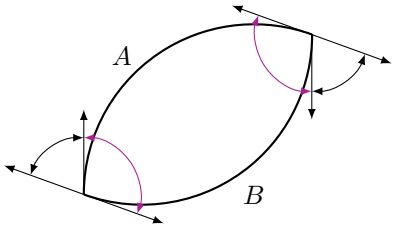
```
repeat 120 [fd 2 rt 1]
rt 60
repeat 120 [fd 2 rt 1]
rt 60
```

oder noch einfacher:

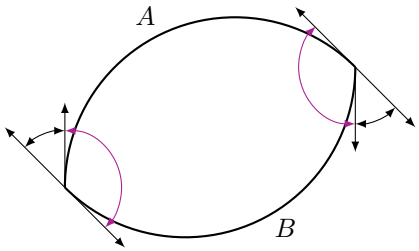
```
repeat 2 [repeat 120 [fd 2 rt 1] rt 60]
```

Probiere es aus.

Jetzt könnten wir uns wünschen, schmalere Blätter (die Teile A und B sind kürzer) oder breitere Blätter (die Teile A und B sind länger) zu zeichnen.



ANTEIL = 120°



ANTEIL = 135°

Dazu können wir wieder einen Parameter verwenden. Nennen wir den Parameter zum Beispiel `:ANTEIL`. Dann berechnen wir die Drehung an der Spitze des Blattes wie folgt:

Bevor wir den Teil B des Blattes zeichnen, soll die Hälfte der ganzen Drehung, das heisst $\frac{360°}{2} = 180°$, gemacht werden. Also ist die Drehung an der Spitze des Blattes

$$180° - \texttt{:ANTEIL}.$$

Damit können wir folgendes Programm im Editor schreiben:

```
to BLATT :ANTEIL
repeat 2 [repeat :ANTEIL [fd 2 rt 1] rt 180-:ANTEIL]
end
```

Probiere dann das Programm aus, indem du die folgenden Aufrufe in die Befehlszeile schreibst:

```
BLATT 20
BLATT 40
BLATT 60
BLATT 80
BLATT 100
```

Was passiert?

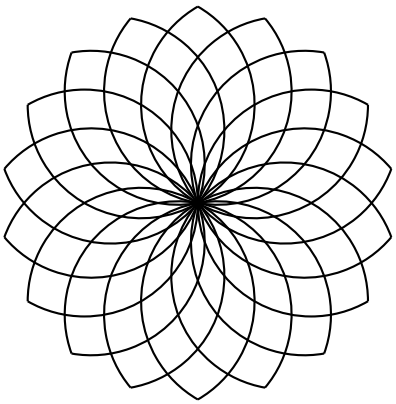
Eine Blume hat viele spitze Blätter

Nun wollen wir BLATT als Unterprogramm nutzen, um Blumen mit spitzen Blättern zu zeichnen.

Aufgabe 65

Zeichne zuerst mit folgendem Programm eine Blume:

```
BLATT 100
rt 20
BLATT 100
rt 20
BLATT 100
....
```



Wie oft musst du die Befehle BLATT und rt 20 wiederholen, um diese Blume vollständig zu zeichnen?

Schreibe das Programm für die Blume in nur einer Zeile mit einem geeigneten **repeat**-Befehl. (Denke dabei daran, dass alle Drehungen rt zwischen den einzelnen Blättern zusammen 360° ergeben müssen.)

Aufgabe 66

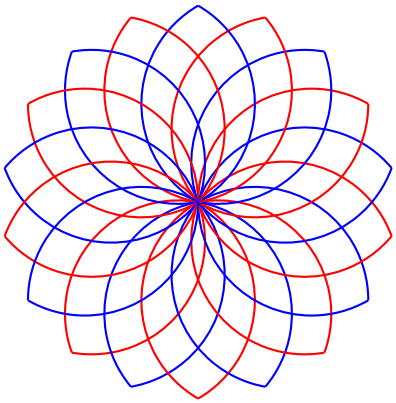
Gib das Programm aus Aufgabe 65 im Editor ein. Nenne das Programm BLUME3. Das Programm soll den Parameter :ANTEIL haben. Was passiert, wenn du BLUME3 60, BLUME3 80 und BLUME3 100 eingibst?

Aufgabe 67

(a) Schreibe ein Programm mit einem Parameter, das die Blume aus der Aufgabe 65 in einer frei wählbaren Farbe zeichnet. Nenne dein Programm `BLUME4`.

(b) Ändere dein Programm zu `BLUME5` nun so, dass die Anzahl der Blätter, die gezeichnet werden, von einem neuen Parameter `:ANZ` festgelegt wird. Denke daran, dass alle Drehungen `rt` zwischen den einzelnen Blättern zusammen 360° ergeben müssen.

(c) Ändere dein Programm `BLUME5` so, dass die Blume nun in zwei frei wählbaren Farben gezeichnet wird. Nenne das neue Programm `BLUME6`.



Aufgabe 68

Im Programm `BLATT` bestimmt der Befehl `fd 2` die Größe des Kreises, aus dem wir den Teilkreis des Winkels `:ANTEIL` schneiden. Diesen Wert 2 können wir auch durch einen Parameter namens `:GR` (Grösse) ersetzen. Schreibe ein Programm

```
BLAETTER :ANTEIL :GR
```

mit den Parametern `:ANTEIL` und `:GR`, mit denen wir den Teilkreis und die Größe einstellen können. Probiere es dann aus mit den folgenden Programmaufrufen:

```
BLAETTER 100 1
BLAETTER 100 1.5
rt 100
BLAETTER 80 2
BLAETTER 80 2.5
```

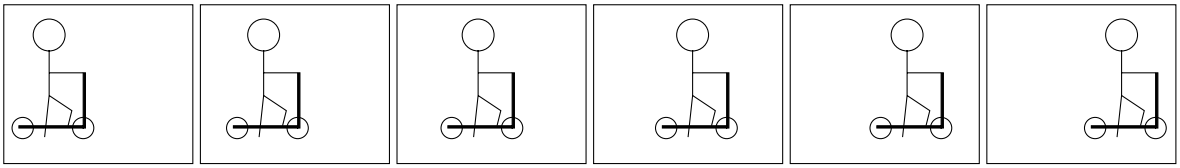
Drehe dann die Schildkröte nach rechts um 80° und wiederhole das obige Programm.

Aufgabe 69

Denke dir andere Fantasiebilder aus.

7 Programmieren von Animationen

Weißt du, wie man Zeichentrickfilme herstellt? Es funktioniert genau gleich wie beim Daumenkino. Man zeichnet zuerst ein paar Bilder, die sich jeweils nur ganz wenig von einander unterscheiden. Im folgenden Bild bewegt sich zum Beispiel der Junge auf dem Kickboard von Bild zu Bild jeweils um ein kleines Stück:

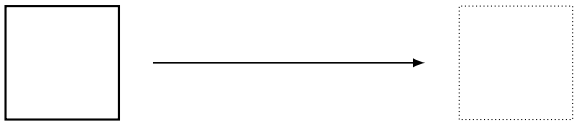


Legt man die Bilder alle übereinander und blättert sie mit dem Daumen schnell durch, hat man das Gefühl, dass der Junge mit seinem Kickboard von links nach rechts fährt. Bewegte Bilder nennt man **Animationen**.

In dieser Lektion lernen wir, wie wir eine Animation mit Hilfe der Schildkröte programmieren können.

Ein Quadrat, das Spuren hinterlässt

In unserer ersten Animation wählen wir eine Figur, die nicht zu schwierig ist und die wir schon lange kennen: Wir werden ein Quadrat von links nach rechts wandern lassen.



Das Programm für das Quadrat kennen wir schon von früher:

```
to QUAD100
repeat 4 [fd 100 rt 90]
end
```

Nachdem das Quadrat einmal gezeichnet worden ist, verschieben wir die Schildkröte ein bisschen nach rechts und zeichnen das Quadrat noch einmal. Dann verschieben wir die Schildkröte wieder nach rechts und zeichnen erneut ein Quadrat. Das wiederholen wir ein paar Mal.

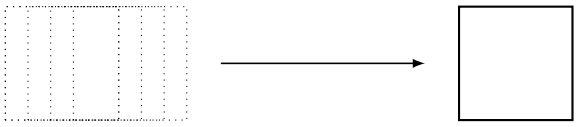
Im folgenden Programm zeichnen wir 120 solcher Quadrate:

```
to QUADLAUF
repeat 120 [QUAD100 rt 90 fd 4 lt 90]
end
```

Aufgabe 70

Schreibe die Programme `QUAD100` und `QUADLAUF` in den Editor und führe `QUADLAUF` aus. Was wird gezeichnet?

Du siehst, dass die Spur *aller* Quadrate gezeichnet wird. Bei einer Animation möchten wir aber nur immer das letzte Quadrat sehen und die Spur löschen.

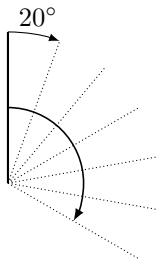


Aufgabe 71

Lass das Quadrat von unten nach oben anstatt von links nach rechts wandern.

Aufgabe 72

Schreibe ein Programm für eine Linie der Länge 20. Verwende dieses Programm, um eine Linie um ihren untersten Punkt im Uhrzeigersinn zu drehen:



Ein Quadrat zeichnen und wieder ausradieren

Um die Spur zu verwischen, müssen wir lernen, Figuren auszuradieren, die wir gerade gezeichnet haben. Dazu muss die Schildkröte statt dem Stift einen Radiergummi benutzen. Mit dem neuen Befehl **`penerase`** oder, viel kürzer, **`pe`** wechselt die Schildkröte vom Stift zum Radiergummi.

Aufgabe 73

Überlege dir, was das Programm `QUAD100 pe QUAD100` macht, ohne es am Computer auszuführen.

Soll die Schildkröte wieder anfangen zu zeichnen, müssen wir es ihr klar mitteilen. Auch dafür gibt es einen neuen Befehl: **`penpaint`** oder, viel einfacher, **`ppt`**. Wir verwenden den neuen Befehl gleich im Programm aus Aufgabe 73.

Das Programm sieht nun wie folgt aus:

```
QUAD100 pe QUAD100 ppt
```

Aufgabe 74

Führe das obige Programm aus. Was passiert? Kannst du es dir erklären?

Das Quadrat muss ein bisschen warten

Wie du beim Lösen in der Aufgabe 74 sicher festgestellt hast, wird das Quadrat nach dem Zeichnen sehr schnell gelöscht. Wir merken nicht einmal, dass überhaupt ein Quadrat gezeichnet worden ist. Bevor wir das Quadrat ausradieren, müssen wir den Computer ein wenig warten lassen.

Dies können wir wie folgt tun:

`wait`	`4`
Wartebefehl	Wartezeit

Aufgabe 75

Probiere das Programm

```
QUAD100 wait 4 pe QUAD100 ppt
```

aus.

Ein Quadrat, das sich von links nach rechts bewegt

Nun sind wir bereit, das Löschen des Quadrats und das Warten in unser Programm QUADLAUF einzubauen:

```
to QUADLAUF
repeat 120 [QUAD100 wait 4 pe QUAD100 rt 90 fd 4 lt 90 ppt]
end
```

Probiere es aus. Wenn dich die Schildkröte beim Zeichnen der Animation stört, dann beginne das Programm mit dem Befehl **hideturtle** (oder kürzer: **ht**), der die Schildkröte verschwinden lässt. Du wirst sogar merken, dass die Animation schneller wird. Beende das Programm mit dem Befehl **showturtle** (oder kürzer: **st**) direkt vor dem **end**. Dadurch wird die Schildkröte wieder sichtbar.

Aufgabe 76

Bewege ein Quadrat der Größe 50×50 nach oben.

Aufgabe 77

Ändere das Programm QUADLAUF so, dass das Quadrat doppelt so schnell nach rechts läuft.

Aufgabe 78

Schaffst du es auch, das Programm QUADLAUF so zu ändern, dass das Quadrat halb so schnell nach rechts läuft?

Aufgabe 79

Ändere das Programm QUADLAUF so, dass sich das Quadrat von rechts nach links statt von links nach rechts bewegt.

Aufgabe 80

Überlege zuerst, was das folgende Programm macht und überprüfe dann deine Vermutung, indem du das Programm ausführst:

```
to QUADLAUF1
ht
repeat 50 [QUAD100 wait 5 pe QUAD100 fd 3 rt 90 fd 3 lt 90 ppt]
QUAD100
st
end
```

Aufgabe 81

Überlege zuerst, was das folgende Programm macht und überprüfe durch seine Ausführung deine Vermutung:

```
to KREISEN
ht
repeat 360 [QUAD100 wait 4 pe QUAD100 fd 5 rt 1 ppt]
QUAD100
st
end
```

Aufgabe 82

Modifiziere das Programm `KREISEN` so, dass sich das Quadrat viermal so schnell bewegt.

Aufgabe 83

Was macht das folgende Programm?

```
repeat 6 [KREISEN]
```

Aufgabe 84

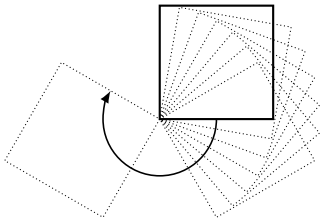
Nimm das folgende Programm

```
to ERDE
repeat 45 [fd 16 rt 8]
end
```

und verwende es, um eine Animation zu zeichnen, in der die Erde auf einer Kreisbahn rund um die Sonne läuft. Wie du die Sonne darstellst, ist deiner Phantasie überlassen.

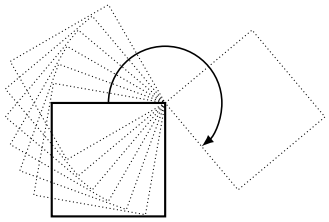
Aufgabe 85

Drehe ein Quadrat im Uhrzeigersinn um seine linke untere Ecke. Die Seitenlänge darfst du selber wählen:



Aufgabe 86

Drehe nun das Quadrat im Uhrzeigersinn um seine rechte obere Ecke:



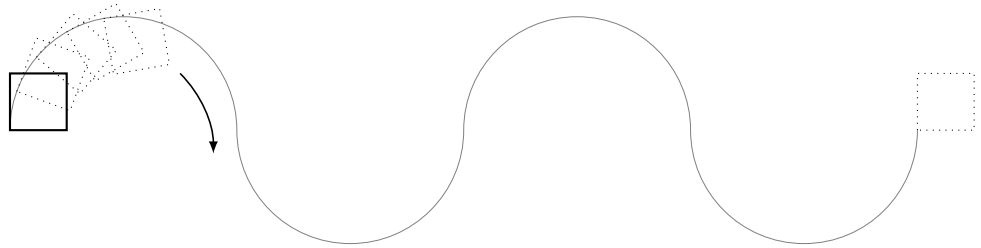
Falls du schon Parameter kennst, kannst du die folgenden Aufgaben bearbeiten.

Aufgabe 87

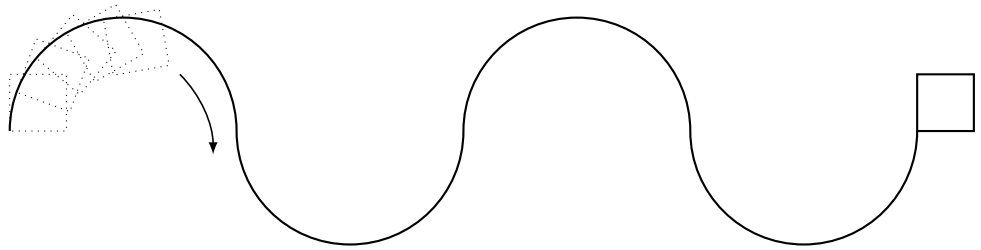
Schreibe ein Programm mit *zwei Parametern*, um ein Quadrat von links nach rechts wandern zu lassen. Ein Parameter soll die Seitenlänge bestimmen, der andere Parameter soll bestimmen, wie schnell sich das Quadrat bewegt.

Aufgabe 88

(a) Lass ein Quadrat auf der unten gezeichneten Bahn laufen, die aus 4 Halbkreisen besteht. Die Seitenlänge des Quadrats soll durch einen Parameter bestimmt werden.

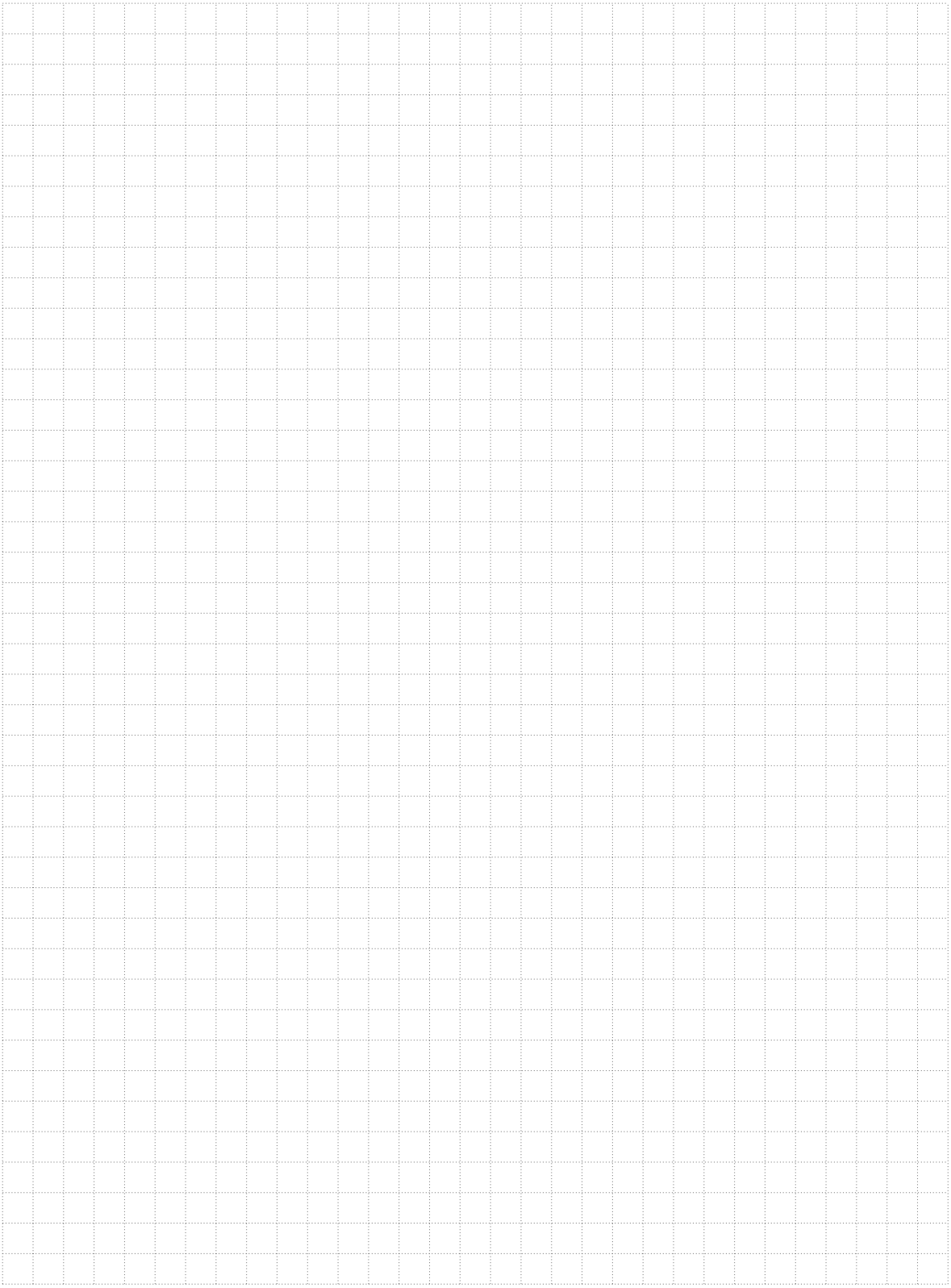


(b) Nun soll die Bahn bei der Bewegung als Spur mitgezeichnet werden.



(c) Kannst du das Programm aus (b) so erweitern, dass auch die Anzahl der Halbkreise durch einen Parameter bestimmt wird?

Meine Notizen



Befehlsübersicht

`fd 100`	100 Schritte vorwärts gehen
`bk 50`	50 Schritte rückwärts gehen
`cs`	alles löschen und neu beginnen
`rt 90`	90 Grad nach rechts drehen
`lt 90`	90 Grad nach links drehen
`repeat 4 [...]`	das Programm in `[...]` wird viermal wiederholt
`pu`	die Schildkröte wechselt in den Wandermodus
`pd`	die Schildkröte wechselt zurück in den Stiftmodus
`setpc 3`	wechselt die Stiftfarbe auf die Farbe 3
`to NAME`	erstellt ein Programm mit einem Namen
`to NAME :PARAMETER`	erstellt ein Programm mit einem Namen und einem Parameter
`end`	alle Programme mit einem Namen enden mit diesem Befehl
`pe`	die Schildkröte wechselt in den Radiergummimodus
`ppt`	die Schildkröte wechselt zurück in den Stiftmodus
`wait 5`	die Schildkröte wartet 5 Zeiteinheiten

Programmieren in LOGO

Professur für Informatik und ihre Didaktik
Universität Trier, H 444
Behringstrasse 21
D – 54296 Trier