



Die Bruno-Schleife



Das Ziel der zweiten Unterrichtseinheit ist es, den Kindern das erste strukturgebende Element, das Prinzip der Schleife, nahezu bringen. Und zwar, indem die Kinder selbst darauf kommen.

Dazu sollen die Kinder, angefangen bei einem Quadrat weitere regelmäßige n-Ecke programmieren. Je höher das n, desto mehr Codezeilen müssen die Kinder nach ihrem aktuellen Kenntnisstand untereinander schreiben.

„Puh, das ist schon die 30ste Zeile Code. Das muss doch einfacher gehen“, werden die Ersten beim 36-Eck stöhnen.

Das Problem ist also gefunden und der Wunsch groß, es zu lösen. Gemeinsam gehen die Kinder auf die Suche und entdecken spielerisch die Möglichkeit, Wiederholungen in ihr Skript einzubauen. Wie von alleine verfügen sie nun über neues Wissen, mit dessen Hilfe sie schon die wunderbarsten Muster programmieren können.



Überblick

Kompetenzen

Zeitaufwand	90 Minuten
Technik	touchfähige Geräte, WLAN (auch am PC mit LAN)
Methoden	Gruppenarbeit, Frage und Antwort, Simulation
Vorkenntnisse	Curriculum 1

Die Schülerinnen und Schüler ...

- festigen ihr Wissen und ihre Umsetzungskompetenz aus der letzten Stunde.
- lernen das Programmierprinzip der Schleife kennen.
- programmieren verschiedene Muster.
- wissen am Ende, wie sie es schaffen, mithilfe der Schleife einen Mäander zu zeichnen

Ablauf

Phase	Aufgabe	Methode	Zeit
Reflexion	Wiederholung des grundlegenden Wortschatzes	F & A	10'
Arbeitsphase I	Das Sechseck	Gruppenarbeit & Simulation	25'
	Regelmäßige n-Ecke		
	Break: Die For-Schleife		15'
Freiarbeit	Eigene Muster	Simulation	10'
Arbeitsphase II	Mäander	Gruppenarbeit	25'
Rück- und Ausblick	Zusammenfassung	Plenum	5'



Unterrichtsverlauf

Vorbemerkung








Die Erfahrung in den Praxiseinheiten von Code your Life zeigt, dass sich jede Klasse unterschiedlich schnell in die Thematiken einfindet. Auch innerhalb der Lerngruppen brauchen manche etwas länger und mehr Übungsphasen, andere sind ganz fix und denken bereits weiter. Auch im Handling mit dem Computer sind die Kinder unterschiedlich stark erfahren. Die Zeitangaben der vorliegenden Unterrichtseinheiten sind demnach nur ungefähre Richtwerte. Geben Sie Ihrer Lerngruppe Zeit, sowohl den Wortschatz als auch das neue Programmierprinzip der Schleife wirklich zu verstehen.

Die Kinder entwickeln nun selbstständig Ideen! Bei der Unterrichtseinheit 2 geht es nun um die sogenannten Schleifen. Mithilfe der Schleifen können Code-Abfolgen beliebig oft wiederholt werden. Das Curriculum ist so aufgebaut, dass die Kinder selbst an den Punkt kommen, sich eine Art Wiederholungsfunktion zu wünschen. Das Selbsterkennen hilft ungemein dabei, das Programmierprinzip zu verstehen. Unterstützen Sie Ihre Schülerinnen und Schüler dabei, Dinge selbst herauszubekommen.

Phase 1 | Reflektion

Gelerntes wiederholen! Nutzen Sie den Anfang der Stunde, um mit den Kindern die erlernten Dinge aus der letzten Unterrichtseinheit zu wiederholen. Fragen Sie die Schülerinnen und Schüler nach den grundlegenden Befehlen der Turtle. Wiederholen Sie ebenso (mündlich) die Aufgaben und Lösungswege der letzten Stunde.

Der Wortschatz noch einmal im Überblick:

 turtle → pen up	Die Turtle nimmt den Stift hoch.
 turtle → pen down	Die Turtle setzt den Stift ab.
 turtle → left turn (90)	Die Turtle dreht sich um 90° nach links.
 turtle → right turn (180)	Die Turtle dreht sich um 180° nach rechts.
 turtle → forward (200)	Die Turtle läuft um 200px nach vorne.
 turtle → back (100)	Die Turtle läuft um 100px rückwärts.
 turtle → set pen color (colors -> red)	Die Turtle wechselt die Farbe auf „rot“.



Phase 2 | Arbeitsphase I

Ziel der Arbeitsphase ist, zu erkennen, wie sich Winkel in einem regelmäßigen n-Eck berechnen lassen und wie man es schafft, dass die Turtle mit nur 2 (bzw. 4) Zeilen Code, ein beliebiges n-Eck zeichnet. Die Kinder werden in dieser Unterrichtseinheit das Programmierprinzip der Schleife kennenlernen.

2.1 Praktisches Wiederholen

Nachdem Sie die Geräte ausgeteilt haben, bzw. die Kinder die Computer hochgefahren haben, lassen Sie zuerst noch einmal ein einfaches Quadrat zeichnen.



Aufgabe:

Die Schildkröte soll ein Quadrat mit einer Kantenlänge von 100 Pixeln zeichnen!

```
function main ()  
  turtle → forward(100)  
  turtle → left turn(90)  
  turtle → forward(100)  
  turtle → left turn(90)  
  turtle → forward(100)  
  turtle → left turn(90)  
  turtle → forward(100)  
  turtle → left turn(90)  
end function
```

CyL 2.1 Quadrat



2.2 Vorbereitung der ersten Aufgabe

Sobald die Schülerinnen und Schüler ein Quadrat auf ihrem Bildschirm haben, können Sie die nächste Aufgabe nennen.

!! Aufgabe:

Die Schildkröte soll ein regelmäßiges 6-Eck mit einer Kantenlänge von 100 Pixel zeichnen.

Zeichnen Sie ein regelmäßiges 6-Eck an die Tafel, bzw. lassen Sie einen Schüler oder eine Schülerin ein 6-Eck zeichnen. Überlegen Sie gemeinsam, was sie an den Codezeilen des Quadrats verändern müssen, um ein 6-Eck zu bekommen. Folgende Fragen bieten sich an:

- Muss man die Seitenlänge ändern? (Antwort: Nicht unbedingt.)
- Muss die Anzahl der Seiten verändert werden? Wenn ja, auf wieviel? (Antwort: Auf 6.)
- Muss der Winkel verändert werden? (Antwort: Ja)
- Und muss sich die Turtle immer gleich weit drehen? (Antwort: Ja)
- Um wieviel Grad muss sich die Turtle an jeder Ecke drehen? (Antwort: 60 Grad)

Je nach Kenntnisstand der Schülerinnen und Schüler wird es ihnen leicht oder schwer fallen, selbst auf die Winkelgröße zu kommen.

i Hinweis! Helfen Sie den Schülern dabei, indem sie wieder die Schildkröte simulieren:

Stellen Sie sich vor die Schüler und laufen Sie das Quadrat ab. Zählen Sie dabei in jeder Ecke mit „1x90 Grad, 2x90 Grad, 3x90 Grad, 4x90 Grad“, bis Sie wieder an der Ausgangsposition angekommen sind. Fragen Sie die Schüler:

- Wieviel Grad habe ich mich insgesamt gedreht? (Antwort: 360 Grad)

Stellen Sie sich vor die Schüler und laufen Sie das Quadrat ab. Zählen Sie dabei in jeder Ecke mit „1x90 Grad, 2x90 Grad, 3x90 Grad, 4x90 Grad“, bis Sie wieder an der Ausgangsposition angekommen sind. Fragen Sie die Schüler:

- Wieviel Grad habe ich mich jetzt insgesamt gedreht? (Antwort: Wieder 360 Grad)

Wiederholen Sie dies gegebenenfalls für ein 8-Eck. Zeigen Sie ihren Schülern so, dass sich die Turtle bei einem regelmäßigen n-Eck immer um 360 Grad dreht.



Nächste Frage:

- Um wieviel Grad muss ich mich bei einem Sechseck nun jedes Mal drehen?
(Antwort: 60 Grad)

Die Schüler werden vielleicht raten und z.B. 45 Grad sagen. Geben Sie die Lösung nicht vor, sondern helfen Sie den Schülern, hier selbst auf die notwendige Rechnung zu kommen. Nehmen Sie sich die Zeit und lassen Sie die Schüler nachdenken, so dass sie den Lösungsweg tatsächlich verstehen.

Geben Sie als Hilfestellung und für einen besseren Überblick ein Tafelbild und erstellen Sie mit den Kindern folgende Tabelle:

Anzahl Ecken	Winkelgrad der Ecken	Gradzahl gesamt
4	90	360
6	60	360
8	45	360
12	30	360
36	10	360

2.3 Das regelmäßige 6-Eck

Nun geht es an die Umsetzung des 6-Ecks auf dem Bildschirm. Geben Sie den Kindern den Hinweis, dass sie die Codezeilen des Quadrates nicht löschen müssen, sondern durch die besprochenen Veränderungen aus dem Quadratprogramm eins für 6-Ecke machen können.



!! Aufgabe: Die Schildkröte soll ein regelmäßiges 6-Eck mit einer Kantenlänge von 100 Pixel zeichnen.

```
function main ()
  turtle → forward(100)
  turtle → left turn(60)
  turtle → forward(100)
  turtle → left turn(60)
  turtle → forward(100)
  turtle → left turn(60)
  turtle → forward(100)
  turtle → left turn(60)
  turtle → forward(100)
  turtle → left turn(60)
  turtle → forward(100)
  turtle → left turn(60)
end function
```

CyL 2.2.3 6-Eck

2.4. Weitere n-Ecke

Je nachdem wie schnell die Teams sind, können Sie weitere Aufgaben geben:

!! Aufgabe:
Die Schildkröte soll ein regelmäßiges 8-Eck, 12-Eck, 36-Eck zeichnen.

Spätestens beim 36-Eck werden die Schülerinnen und Schüler aufstöhnen, dass das anstrengend ist. Wenn die Kinder dies selbst bemerken und bemängeln, sind sie bereit,



das erste Programmierprinzip kennenzulernen. Dies ist genau der richtige Moment, um die Schleife einzuführen.

2.5 Break - Die Schleife

Greifen Sie diese leichte Ermüdungserscheinung und Frustration der Kinder auf und fragen Sie, was Sie sich an dieser Stelle wünschen würden, z.B.

- Was wäre toll zu haben? (Antwort: Eine Wiederholungsschleife)

Bitte Sie zwei Schüler, z.B. Bruno und Tara nach vorne und simulieren Sie mit ihnen das Programm. Sie sind die Schildkröte. Bruno ist der Befehlsgeber, Tara der Zähler.



Brunos und Taras Aufgabe ist es nun, dass die Schildkröte, also Sie, ein 6-Eck läuft. Dazu sagt Bruno den Befehl:

```
turtle → forward (100)
```

```
turtle → left turn (60)
```

Und Tara zählt diese beiden Befehle bis Sie als Turtle diese 6x durchgeführt haben. Dazu klopft sie Bruno bei jeder Zahl auf die Schulter. Wenn das 6-Eck geschafft ist, sagt Tara „Ende“.

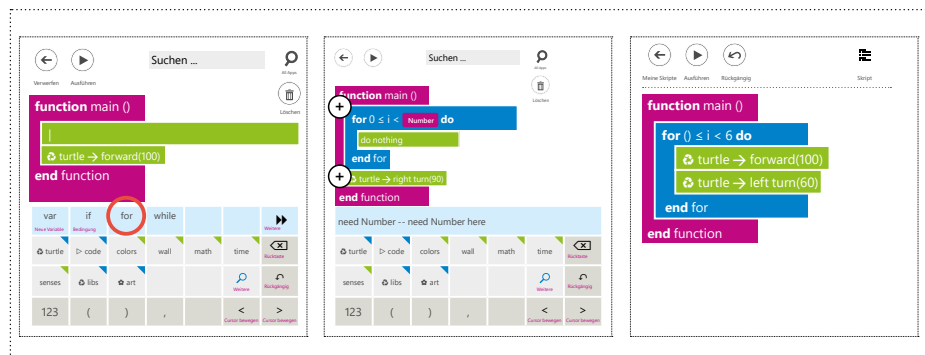


Malen Sie nun die Schleife folgendermaßen an die Tafel:

```
bruno tara ()  
  turtle → forward(100)  
  turtle → left turn(60)  
tara
```

i Hinweis! Wir haben gute Erfahrungen gemacht, an dieser Stelle den Namen For-Schleife etwas spielerischer zu gestalten und die Schleife nach einem Schüler zu benennen, wie in unserem Beispiel Bruno. So heißt die For-Schleife von nun an Bruno-Schleife. Sie werden sehen, dass die Kinder sich dies sehr leicht merken können und Bruno wird es stolz machen.

Erklären Sie den Kindern, dass dies nun die Bruno-Schleife ist, die in Wirklichkeit For-Schleife heißt. Lassen Sie die Schülerinnen und Schüler auf den Tastaturfeldern der Turtle nach dem passenden Befehl suchen.



Nun haben Sie das erste Programmierprinzip eingeführt. Lassen Sie nun alle Kinder die verschiedenen n-Ecke mit der Bruno-Schleife ausprobieren.

😊 Lösungsbeispiel für ein 12-Eck

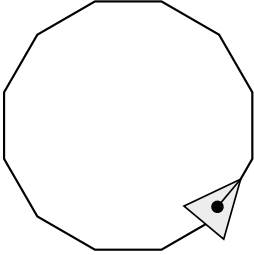
←
▶
↶

☰

Meine Skripte
Ausführen
Rückgängig
Skript

```

function main ()
  for () ≤ i < 12 do
    🔄 turtle → forward(50)
    🔄 turtle → left turn(30)
  end for
end function
          
```



Phase 3 | Freiarbeit

3.1. Muster programmieren

Um das Schleifenprinzip zu verinnerlichen, schieben Sie eine Phase der Freiarbeit ein. So können die Schülerinnen und Schüler das neu gefundene Programmierprinzip eigenständig einüben und Routine im Umgang mit den Codezeilen bekommen. Darüber hinaus fördert die Freiarbeit die Motivation, tiefer einzusteigen. Es werden ganz verschiedene Formen und Figuren entstehen: Kreisbilder, Sterne, Zick-Zack-Linien usw.

!! Aufgabe:

Denkt euch verschiedene Muster aus, die ihr mithilfe der Bruno-Schleife programmieren wollt.

An dieser Stelle möchten wir Ihnen noch einen „Geheimbefehl“ mit auf den Weg geben. Eventuell werden die Kinder Sie nämlich fragen, ob man die Ausführung der Turtle-Bewegungen nicht beschleunigen kann, um nicht immer so lange auf das fertige Bild warten zu müssen. Ja, kann man, mithilfe folgender Befehle:



 turtle → fast


fertiges Ergebnis wird sehr schnell angezeigt

 turtle → set speed (400)

Geschwindigkeit wird festgelegt

Besonders schöne Effekte erzielen die Kinder mit dem Befehl für zufällige Farbwahl:

 set pen color (colors → random)

 **Hinweis!** Je nach Aufmerksamkeits- und Aufnahmefähigkeit der Kinder können Sie die Freiarbeit zeitlich noch ausweiten. Sie können entscheiden, ob Sie die nun folgende Arbeitsphase II in die nächste Unterrichtsstunde verlagern möchten, um mit neuer Energie zu starten.

Phase 4 | Arbeitsphase II

3.2 Mäander programmieren

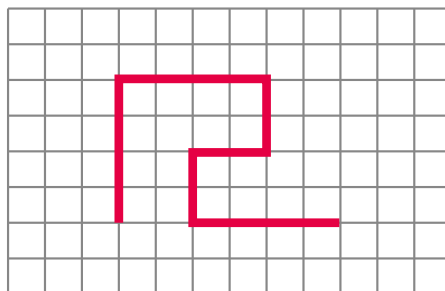
Erinnern Sie die Kinder nun an die Mäander. Stellen Sie dazu die Frage:

- Wozu haben wir heute die Bruno-Schleife kennengelernt? Was war die letzte Hausaufgabe? (Antwort: Mäander mitbringen.)

Lassen Sie die Schülerinnen und Schüler die Zuhause ausgedruckten Bilder herausholen. Erörtern Sie mit den Kindern, worum es sich bei einem Mäander handelt.

- Was ist ein Mäander? (Antwort: ein Muster, das immer wieder wiederholt wird.)

Zeichnen Sie folgenden Mäander auf die karierten Felder der Tafel und legen Sie als Hilfestellung die Längen fest: Kurze Seiten 50 Pixel, lange Seiten 100 Pixel.





Geben Sie nun die nächste Aufgabe:

!! Aufgabe:

Lasst eure Schildkröte diesen Mäander zeichnen. Nutzt dazu die Bruno-Schleife.

Je nach Zeit, die Ihnen nun noch zur Verfügung steht, lassen Sie noch weitere Mäander in anderen Größen zeichnen.

i Hinweis! Auch hier werden einige Schülerinnen und Schüler schnell auf das Ergebnis kommen, andere benötigen mehr Zeit. Lassen Sie die Kinder sich ruhig gegenseitig helfen und gehen Sie bei Schwierigkeiten die Codezeilen Schritt für Schritt durch.



Lösungsbeispiel des Mäanders











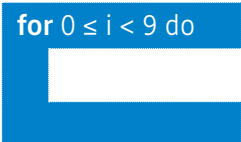
Meine Skripte Ausführen Rückgängig Skript

```
function main ()  
  turtle → pen up  
  turtle → left turn(90)  
  turtle → forward(200)  
  turtle → right turn(90)  
  turtle → pen down  
  for 0 ≤ i < 3 do  
    turtle → forward(100)  
    turtle → right turn(90)  
    turtle → forward(100)  
    turtle → right turn(90)  
    turtle → forward(50)  
    turtle → right turn(90)  
    turtle → forward(50)  
    turtle → left turn(90)  
    turtle → forward(50)  
    turtle → left turn(90)  
    turtle → forward(100)  
    turtle → left turn(90)  
  end for  
end function
```

CyL 2.3.2 Mäander

Phase 4 | Ausblick

Fassen Sie am Ende noch einmal zusammen, was in dieser Unterrichtseinheit alles gemacht wurde. Geben Sie dann einen Ausblick auf das nächste Mal. Im kommenden Curriculum werden die Mäander noch weiter vertieft und ein neues Programmierprinzip wird eingeführt: Merkkästen und Variablen.

Gesamtwortschatz der Turtle (Stand Tutorial 2)	
 turtle → pen up	Die Turtle nimmt den Stift hoch.
 turtle → pen down	Die Turtle setzt den Stift ab.
 turtle → left turn (90)	Die Turtle dreht sich um 90° nach links.
 turtle → right turn (180)	Die Turtle dreht sich um 180° nach rechts.
 turtle → forward (200)	Die Turtle läuft um 200px nach vorne.
 turtle → back (100)	Die Turtle läuft um 100px rückwärts.
 turtle → set pen color(colors -> red)	Die Turtle wechselt die Farbe auf „rot“.
 turtle → fast	Die Turtle zeigt gleich das fertige Bild.
 turtle → set speed (400)	Die Turtle läuft in vorgegebener Geschwindigkeit.
 turtle → set pen color(colors->random)	Die Turtle wechselt die Farbe nach dem Zufallsprinzip.
<pre>for 0 ≤ i < 9 do</pre> 	Mit der For-Schleife, auch Bruno-Schleife genannt, können Befehle beliebig oft wiederholt werden.