Supporting Information

Transparente Teilchensimulation im Chemieunterricht für das vertiefende Verständnis chemischer Prozesse unter Nutzung des Computational Thinking

T. Kraska

Institut für Physikalische Chemie, Department für Chemie, Naturwissenschaftliche Fakultät, Universität zu Köln, Greinstraße 4-6, 50939 Köln

E-Mail: t.kraska@uni-koeln.de https://van-der-waals.pc.uni-koeln.de

Doi:

Inhalt

A) Liste der Programme

.py TigerJython: https://tigerjython.ch/ .sb3 Scratch: https://scratch.mit.edu/

1) Berechnung der Einstellung des chemischen Gleichgewichts $A \rightleftharpoons B$ mit der analytischen Lösung der Differentialgleichung. Dies ist die exakte Lösung, die man zur Überprüfung der Simulationsergebnisse nutzen kann.

`P0-analytisch.py`

2) Dynamische Simulation der Einstellung des Gleichgewichts $A \rightleftharpoons B$. Dieses Programm entspricht Abb. 1 im Haupttext.

`P1-Abb1-chem-Glg-dyn.py`

3) Stochastische Simulation der Einstellung des Gleichgewichts $A \rightleftharpoons B$ für $K = 1$. Dieses Programm entspricht Abb. 2 im Haupttext.

`P2-Abb2-chem-Glg-stoch.py`

4) Stochastische Simulation der Einstellung des Gleichgewichts $A \rightleftharpoons B$ für eine variable Gleichgewichtskonstante K. Dieses Programm entspricht Abb. 4 im Haupttext.

`P3-Abb4-chem-Glg-stoch-K.py`

5) Random Walk auf einem quadratischen Gitter. Dieses Programm entspricht Abb. D1 in Kapitel D.

`P4-Abb-D1-RW-Gitter.py`

6) Random Walk auf einem quadratischen Gitter mit graphischer Ausgabe sowie mit Mittelung über viele Konfigurationen. In Abb. D1 in Kapitel D ist eine mit diesem Programm erzeugte Konfiguration abgebildet.

`P5-RW-Gitter-Graphik.py`

7) Scratch-Programm für den RW auf einem quadratischen Gitter. Dieses Programm entspricht Abb. D3 in Kapitel D (s. a. Kapitel F).

`P6-D3-RW-Gitter.sb3`

8) Scratch-Programm für den RW auf einem quadratischen Gitter mit Mittelwertberechnung des quadratischen End-End-Abstands (s. Kapitel F).

`P7-RW-Gitter-R2Mittelwert.sb3`

B) Umsetzung von Simulationen für die chemische Kinetik im Unterricht

Die Grundlagen der chemischen Kinetik und des chemischen Gleichgewichts nehmen in der Einführungsphase (1. Jahr der Oberstufe) großen Raum ein. Nach der Durchführung einiger Lernendenversuche wie die Reaktion von Zink, Magnesium oder Calciumcarbonat mit Salzsäure, bei denen ein gasförmiges Reaktionsprodukt in einem Kolbenprober aufgefangen wird, wird die Reaktionsgeschwindigkeit über das Steigungsdreieck und die Ableitung eingeführt. Die Anwendung des Steigungsdreiecks auf benachbarte Datenpunkte ist die einfachste Form der numerischen Ableitung von Messwerten. Allerdings wird in diesem Stadium, die Reaktionsordnung ausdrücklich *nicht* thematisiert (es handelt sich um Reaktionen pseudoerster Ordnung, die diffusionskontrolliert sind). An dieser Stelle geht es bei der Auswertung lediglich darum, die Reaktionsgeschwindigkeit zu definieren und zu berechnen. Gleiches gilt für die dann folgende stochastische Simulation einer chemischen Reaktion. Eine Differenzierung zwischen Reaktionen 1. und 2. Ordnung kann im folgenden Schuljahr bei der Besprechung der nukleophilen Substitution erfolgen. Dazu kann man zwei unterschiedliche Spiele durchführen wie in einer früheren Arbeit erläutert [B1].

Den Lernenden wurde die Durchführung des Spiels erläutert. Ihre Aufgabe war es, das Spiel zu spielen und anschließen eine strukturierte Abfolge von Anweisungen zu erstellen. In Abb. B1 ist eine typische Lösung abgebildet. Zusätzlich wurde ein leerer Programmablaufplan (PAP, s. Kapitel G) zur Verfügung gestellt, den sie auf Grundlage der Anweisungsliste ausfüllen sollten. Da PAPs im Chemieunterricht nicht bekannt sind, wurden die Lernenden aufgefordert die Bedeutung der Symbole zu recherchieren. Die Symbolik eines PAPs erwies sich für die Lernenden als gut nachvollziehbar und intuitiv. Hier sind Lernende, die Informatik belegen, zwar im Vorteil, die Aufgabe war aber auch für Lernende ohne entsprechende Vorkenntnisse lösbar. In Abb. B2 ist ein Beispiel abgebildet. Die nächste Aufgabe war es, die Daten als Graphen mit einer Tabellenkalkulation aufzutragen. Ein Beispiel findet sich in Abb. B3.

Im nächsten Schritt wurde der entsprechende Code zur Verfügung gestellt. Bei Lernenden, die bislang keine Erfahrung mit Code haben, war das zunächst kryptische Erscheinungsbild ungewohnt. Die Erarbeitung der Beziehung zwischen dem PAP und dem Code führte jedoch dazu, dass die Lernenden den Code nachvollziehen konnten. Hierbei kann man Gebrauch von der englischen Sprache der Befehle machen und sprachlich die Bedeutung einer Befehlszeile erarbeiten. Es tauchen aber auch Symbole auf, die sich nicht intuitiv erschließen, wie der Befehl für das Quadrieren (`m=n**2`) oder für die Erhöhung einer Variable um eine Zahl (`i=i+1`). Für Lernende, die über keine Vorkenntnisse in Informatik verfügten, war auch das zweifach indizierte Feld (Array) `Feld[][]` ungewohnt sowie die Verknüpfung der Koordinaten eines Spielfeldes mit seinem Zustand, d.h. `Feld[ix][iy]=1` bedeutet, dass ein Stein auf dem Feld `ix` und `iy` liegt. Ist der Wert `Feld[ix][iy]=0`, dann liegt dort kein Stein.

Ablaufplan:

- 6×6 Kästchenfeld aufzeichnen und 36 Papierkügelchen anfertigen
- 1 Papierkügelchen in jedes Kästchen legen
- Zwei Würfel nehmen 1 x-Achse, 1 y-Achse: 

- Möglichkeit 1: Auf dem Feld ist ein Stein → Stein entfernen → Produktmenge: +1 und Edukmenge: -1
  Möglichkeit 2: Auf dem Feld ist kein Stein → Anzahl an Würfen: +1 ; Produkt- und Edukmenge: ±0
- Tabellencalculation:

A	B	C ...
Anzahl an Würfen	Edukmenge	Produktmenge

- Diagramm anfertigen

Abb. B1: Lernendenlösung zur Erstellung einer strukturierten Spielanleitung als Pseudocode.

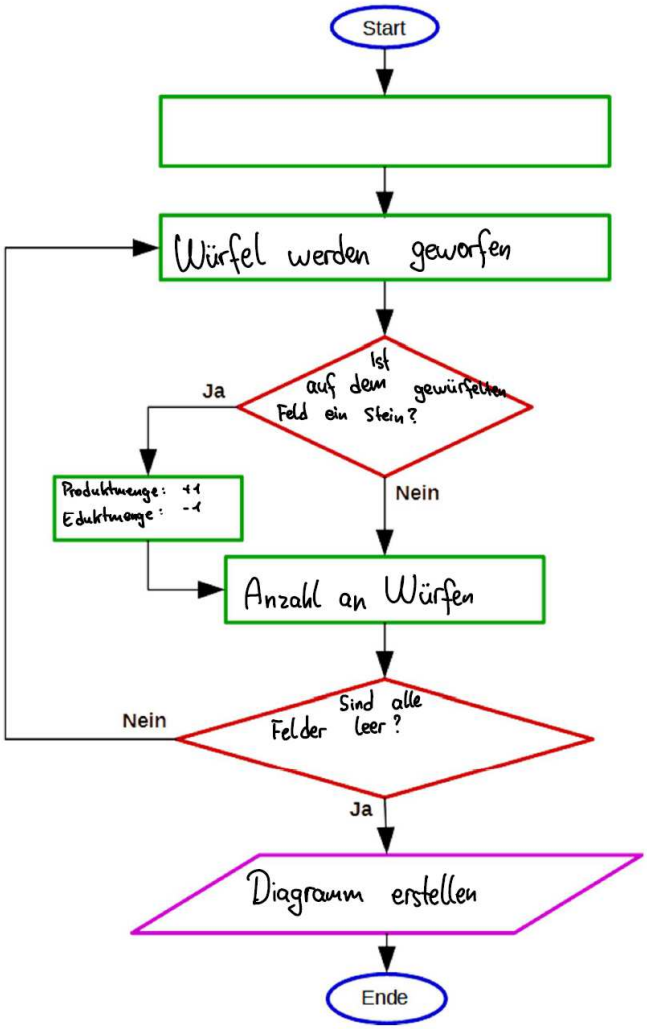


Abb. B2: Lernendenlösung für das Ausfüllen des Programmablaufplans.

	A	B	C	D
1	Würfe	Koordinaten	Entfernt	Geblieben
2	0			36
3	1	4 / 3	1	35
4	2	1 / 1	1	34
5	3	6 / 6	1	33
6	4	4 / 5	1	32
7	5	5 / 2	1	31
8	6	3 / 4	1	30
9	7	2 / 3	1	29
10	8	1 / 2	1	28
11	9	1 / 3	1	27
12	10	6 / 2	1	26
13	11	4 / 3	0	26
14	12	4 / 6	1	25
15	13	3 / 3	1	24
16	14	3 / 6	1	23
17	15	1 / 3	0	23
18	16	6 / 6	0	23
19	17	4 / 5	0	23
20	18	5 / 1	1	22
21	19	6 / 6	0	22
22	20	2 / 6	1	21

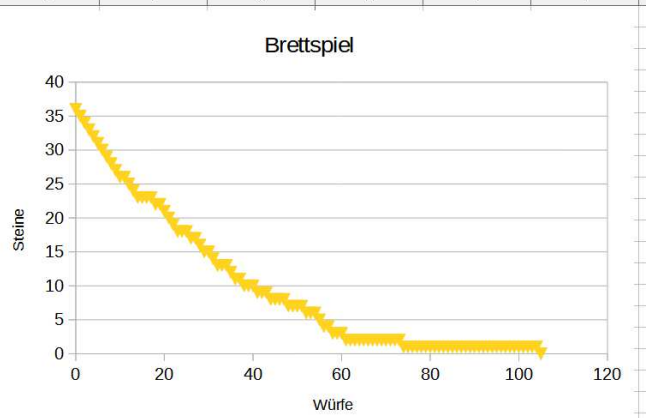


Abb. B3: Lernendenlösung zur Dokumentation des Spiels mit einer Tabellenkalkulation. Die Spalten B und C sind nicht erforderlich, sie wurde von der Gruppe ergänzt.

Durch diese Vorgehensweise ist es möglich, dass auch für Lernende ohne informatische Vorerfahrung ein Simulationscode nachvollziehbar wird, auch wenn eine selbständige Erstellung des Codes zunächst nicht möglich ist. Die unterschiedlichen Vorgehensweisen ermöglichen auch eine Differenzierung im Unterricht je nach Vorkenntnissen. Zusätzlich wurde ein Programm mit graphischer Ausgabe zur Verfügung gestellt. Die Lernenden konnten in den Programmcode eingreifen, z.B. die Zahl der Felder und Steine erhöhen und so das System forschend untersuchen. Wichtig ist, dass man kontinuierlich den Bezug der Simulation zum chemischen System sprachlich aufrechterhält und sich die Simulation davon nicht entkoppelt.

Die iterative Abarbeitung mit einer transparenten Simulation kann das Verständnis der mathematischen Gleichungen ermöglichen und erleichtern [B2]. Wie eine Simulation es ermöglicht durch „leidvolles“ Erleben einen mathematischen Ausdruck zu erfassen, wurde bereits erläutert [B1]: Wenn Lernende das stochastische Brettspiel zur Kinetik erster Ordnung durchführen, dann entfernen sie zu Beginn in kurzen Abständen Steine vom Brett. Wenn jedoch nur noch wenige Steine auf dem Spielbrett liegen, wird das Spiel für einige Lernende in konstruktiver Weise langweilig. Für andere wird es besonders spannend, da sie fieberhaft beim Würfeln der Koordinate entgegenfiebern, auf der noch ein Stein liegt, und das Entfernen des letzten Steines als Zieleinlauf feiern. Vermutlich führt dieses emotionale Erfahren dazu, dass die Lernende die chemischen Vorgänge im Anschluss mit der entsprechenden Differential-gleichung

$$N'(t) \sim N(t) \quad \text{oder} \quad \frac{\mathrm{d}N(t)}{\mathrm{d}t} \sim N(t) \quad \text{oder} \quad \frac{\Delta N(t)}{\Delta t} \sim N(t)$$

zwanglos in Verbindung setzten konnten. Denn natürlich ist die Änderungsrate d.h. die Reaktionsgeschwindigkeit $N'(t)$ umso höher, je mehr Steine $N(t)$ auf dem Brett vorhanden sind. Sofern im parallelen Mathematikunterricht die Exponentialfunktion behandelt wurde, können die Lernenden zur Lösung dieser Differentialgleichung aktiv beitragen [B1].

[B1] Kraska, T. (2021). CHEMKON, 28, 112-121.

[B2] Haraldsrud, A., Odden, O.B. (2023) J. Chem. Educ. 100, 1739-1750.

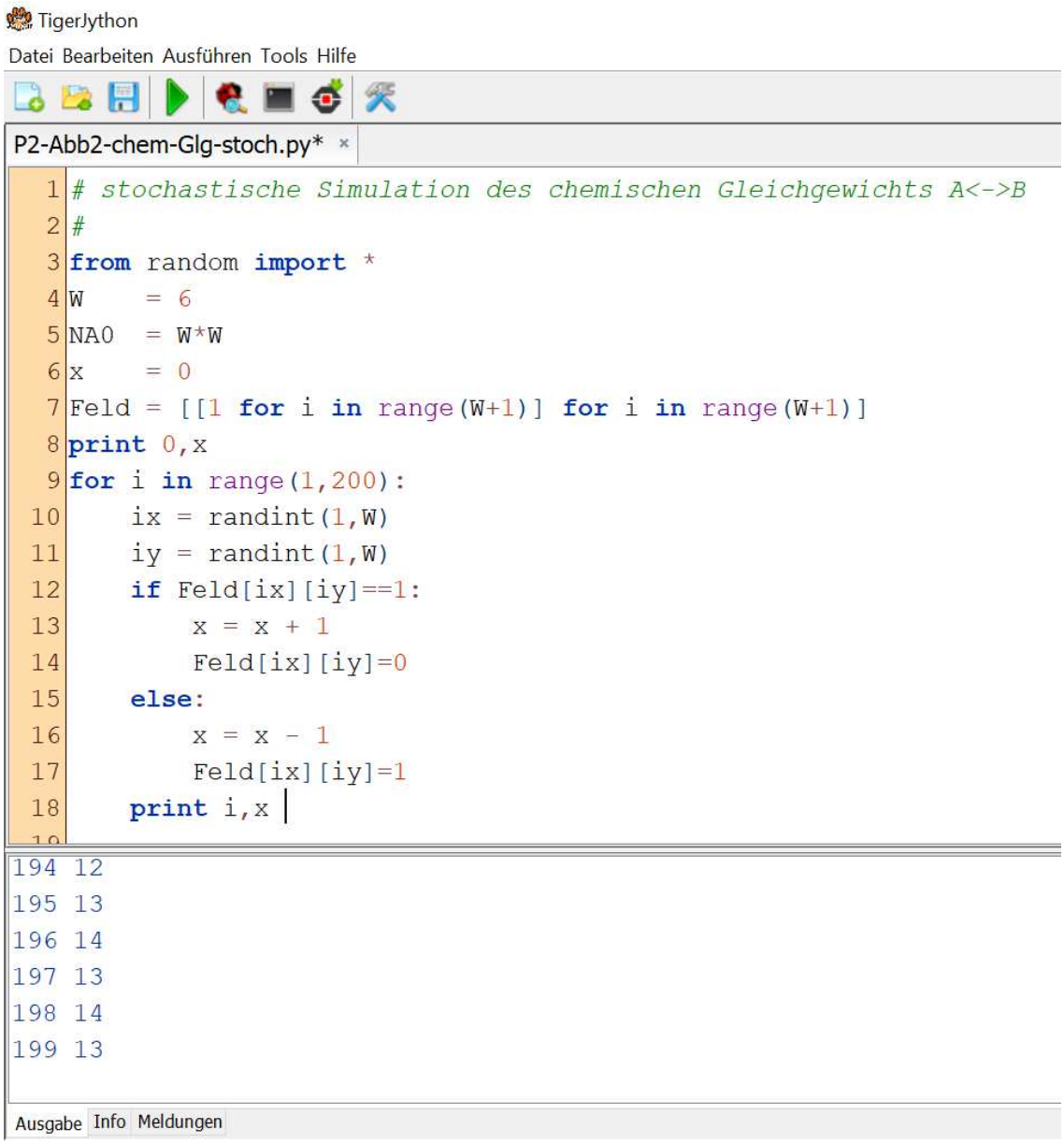
C) Verwendung von TigerJython

TygerJython https://tigerjython.ch/ kann heruntergeladen werden oder in einer Web-Anwendung genutzt werden. Nach dem Start von TigerJython wird über den Menüpunkt „Datei“ der Programmcode geladen. Durch Klick auf den grünen Pfeil wird das Programm gestartet. Die Textausgabe erfolgt in dem unteren Fenster. Diese Ausgabe kann man per Copy-Paste zum Beispiel in eine Tabellenkalkulation kopieren.

Programme mit einer graphischen Ausgabe öffnen ein Graphik-Fenster.

Wichtig bei TigerJython ist die Einrückung mit der TAB-Taste. Die eingerückten Programmteile hängen von der nicht-eingerückten Zeile darüber ab. So werden in dem Beispiel unten die Zeilen 10 bis 18 200mal in der `for`-Schleife wiederholt. Die Zeilen 13 und 14 werden nur ausgeführt, wenn die Bedingung in Zeile 12 erfüllt ist.

Hinweis: „==“ ist ein vergleichendes Gleichzeichen; „=“ ist ein zuweisendes Gleichzeichen



D) Untersuchung der Gestalt von Polymermolekülen im Unterricht

Eine weitere stochastische Simulation, die sich leicht umsetzen lässt, ist die Simulation der Struktur von Polymermolekülen mit flexiblen Bindungen mit dem Random Walk (RW) Modell. Dies ist relevant für die Struktur-Eigenschaftsbeziehungen sowie für die Gummielastizität. Wenn die Bindungen vollständig flexibel sind, dann ist die Ausrichtung der Segmente eines Polymers im Raum nur vom Zufall abhängig. Der RW stellt eine Abfolge von Verschiebungen in zufälliger Richtung dar. Für ein Polymer mit N Segmenten werden N solcher Verschiebungen durchgeführt, so dass jede Verschiebung der Ausrichtung eines Segments des Polymers entspricht. Als Maß für die Gestalt des Polymers kann man den mittleren quadratischen End-End-Abstand berechnen: $\langle R^2 \rangle = (R_1^2 + R_2^2 + \cdots + R_k^2)/k$. Für das RW Modell liefert die Theorie für eine Segmentlänge $\ell = 1$ die Gleichung $\langle R^2 \rangle = N$.

```
1  from random import *
2  from math import *
3  Laenge = 20
4  Konfigurationen = 100
5  R2Sum = 0
6  for n in range(0,Konfigurationen):
7      x = 0
8      y = 0
9      for i in range(0,Laenge):
10         richtung = randint(1,4)
11         if richtung==1:
12             x = x + 1
13         if richtung==2:
14             x = x - 1
15         if richtung==3:
16             y = y + 1
17         if richtung==4:
18             y = y - 1
19     R2 = x**2 + y**2
20     R2Sum = R2Sum + R2
21 R2Mw = R2Sum/Konfigurationen
22 print "<R^2> = ",R2Mw
```

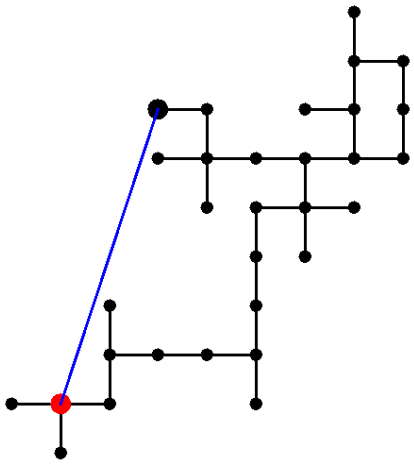


Abb. D1: TigerJython Code zur Simulation des RW auf einem quadratischen Gitter sowie eine graphische Ausgabe für 50 Schritte. Schwarzer Punkt: Anfangspunkt im Ursprung; roter Punkt: Endpunkt; blaue Linie: End-End-Abstand. (Code P4 und P5 in den SI, s. Kapitel A).

Das RW Modell lässt sich am einfachsten auf einem zweidimensionalen quadratischen Gitter simulieren. Dazu wird der Anfang des Moleküls in den Ursprung des Koordinatensystems gelegt und per Zufall eine der vier Richtungen ausgewählt. In diese Richtung wird das Segment dann gelegt und der Vorgang N mal wiederholt. In dem Programm in Abb. D1 wird in der inneren Schleife von Zeile 9 bis 18 ein RW berechnet, indem für jedes Segment eine der vier Richtungen per Zufall ausgewählt wird und die Koordinaten entsprechend um eins erhöht oder erniedrigt werden.

In dem Programm in Abb. D3 ist der gleiche Algorithmus in der graphischen Programmiersprache Scratch [D1] umgesetzt. Eine detaillierte Erläuterung des Programmcodes und Hinweise zu dessen Erstellung sind in Kapitel F zusammengestellt. Scratch-Programme werden aus graphischen Befehlsblöcken mit der Maus auf dem Bildschirm zusammengesetzt. Dies ist intuitiv und wird im Informatikunterricht in der Unterstufe oder auch im NW-Unterricht [D2] eingesetzt. Allerdings ist Scratch für Berechnungen weniger geeignet. Der entsprechende textorientierte TigerJython-Code ist deutlich kürzer und übersichtlicher (s. Kapitel E). Auf der anderen Seite wird bei Scratch die graphische Ausgabe automatisch mitgeliefert, während bei TigerJython eine graphische Ausgabe hinzugefügt werden muss (es sei denn man verwendet die Turtle Graphics Bibliothek, die aber in der Regel die Berechnung zu sehr verlangsamt).

Die Auswertung hinsichtlich der resultierenden Gestalt der Polymermoleküle auf Grundlage des quadratischen End-End-Abstandes können sich Lehrende anhand von zwei speziellen Konfigurationen plausibel machen. Die eine ist repräsentativ für den in der Simulation erhaltenen Wert $\langle R^2 \rangle = N$, der den geknäulten Konfigurationen entspricht. Die andere ist der gestreckte Fall für den $R^2 = N^2$ gilt. Die Erstellung dieser Konfigurationen für gegebenes R^2 kann man als inverse Aufgabe (N und R^2 vorgeben) an die Lernenden geben. In Abb. D2 sind diese beiden Fälle dargestellt. Für den oberen Fall mit $R^2 = N$ sind natürlich viele weitere Konfigurationen denkbar. Genau dies macht die hohe Wahrscheinlichkeit der Knäulung aus und wird durch die Beobachtung in der Simulation in der Form bestätigt, dass man die gestreckte Konfiguration praktisch nicht erhält. Da die Zahl der Anordnungsmöglichkeiten für einen gegebenen Makrozustand (hier End-End-Abstand) nach Boltzmann mit der Entropie zusammenhängt, spricht man von einem entropischen Effekt.

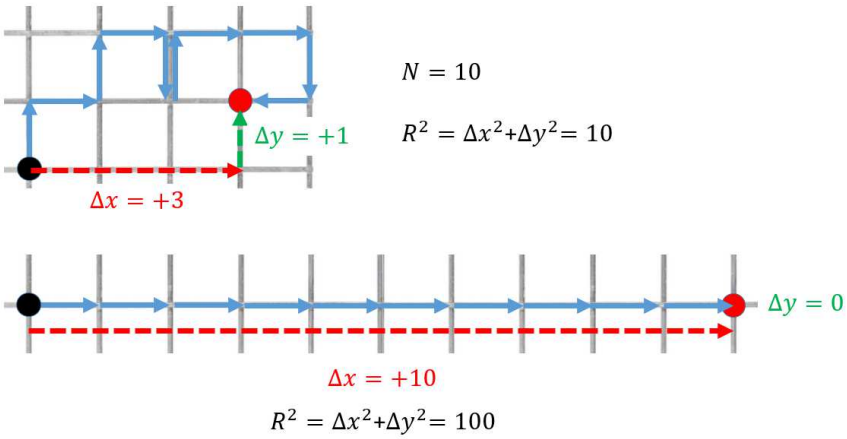


Abb. D2: Ein Random Walk für ein geknäultes Polymer für das $R^2 = N$ gilt, sowie für ein Polymer in der gestreckten Konfiguration mit $R^2 = N^2$.

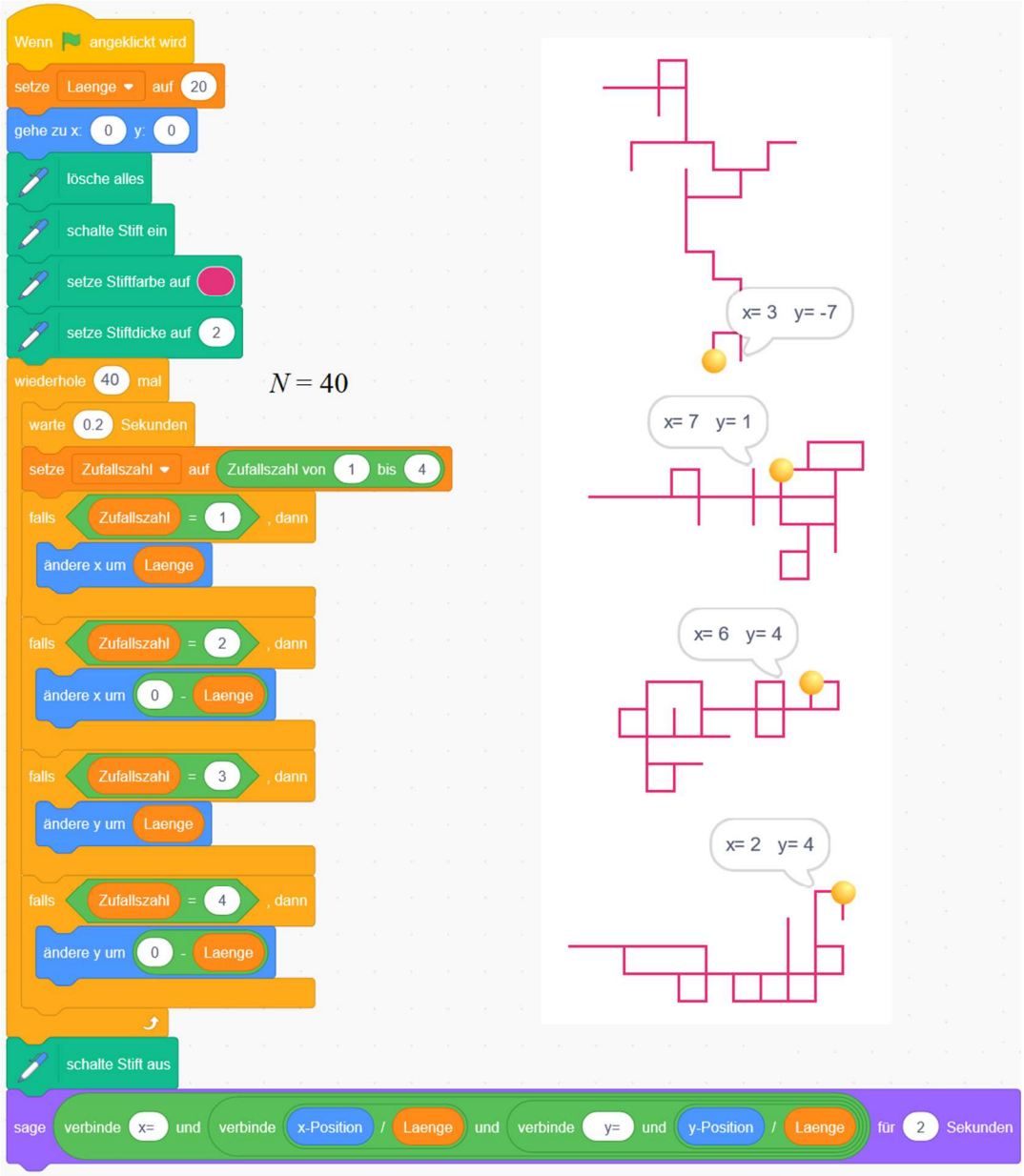


Abb. D3: Scratch Code zur Simulation des RW auf einem quadratischen Gitter. Rechts sind einige Beispiele für $N = 40$ abgebildet. Die Zahlen in den Sprechblasen geben die Verschiebung des Endes relativ zum Anfangspunkt an (Code P6, s. Kapitel A). Der Code ist in Kapitel F erläutert. In Kapitel E werden die dazu erforderlichen Grundlagen zur Bedienung von Scratch erläutert.

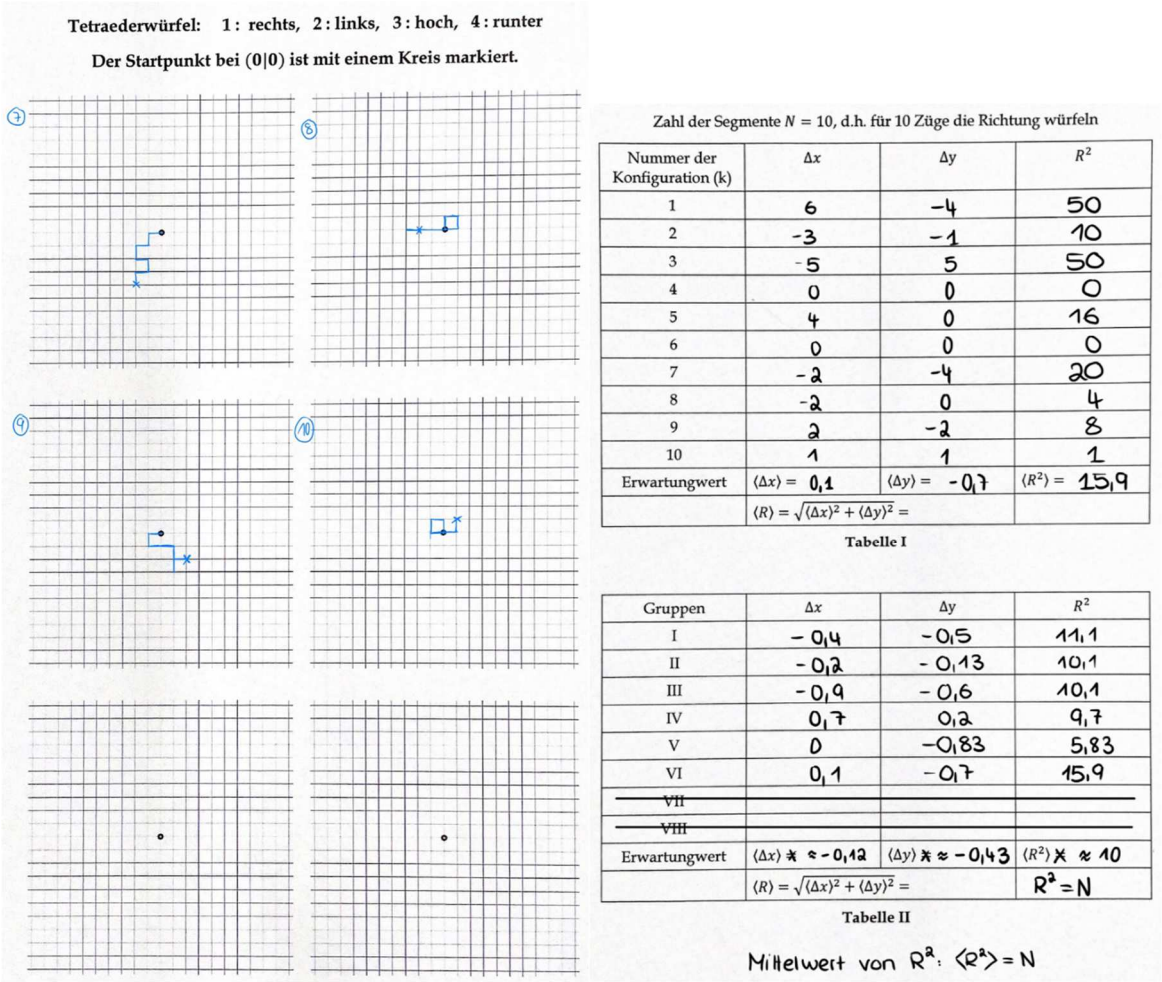
Umsetzung im Chemieunterricht

Die Untersuchung der Gestalt von Polymermolekülen mit flexiblen Bindungen wurde im dritten Jahr der Oberstufe (Q2) im Rahmen des Themas Werkstoffe und Kunststoffe durchgeführt. Der Einstieg verlief über den historischen Streit zwischen H. Staudinger und W. Kuhn um die Frage, ob Polymermoleküle mit frei beweglichen Bindungen gestreckt oder geknäult sind. Mit dem Argument (das auch von einigen SuS vorgebracht wurde), dass in der Näherung von frei beweglichen Bindungen die Richtung einer Bindung nur vom Zufall abhängen kann, wurde der Einstieg in ein stochastisches Spiel möglich. Das System wurde auf ein zweidimensionales

Gitter reduziert und die Richtung mit einem Tetraederwürfel bestimmt. Es war wiederum die Aufgabe für die Lernenden, eine strukturierte Spielanleitung zu erstellen. Die Ergebnisse für $N = 10$ wurden dann auf einem Blatt mit Kästchenmuster notiert wie in Abb. D4 gezeigt. Die numerischen Ergebnisse wurden in einer Tabelle (Abb. D4) notiert. Im Anschluss wurden zur Verbesserung der Statistik mit Hilfe von Tabelle II die Mittelwerte für $\langle \Delta x \rangle$, $\langle \Delta y \rangle$, $\langle R^2 \rangle$ über alle Gruppenergebnisse berechnet. Im Idealfall liegen $\langle \Delta x \rangle$ und $\langle \Delta y \rangle$ nahe bei null und $\langle R^2 \rangle$ stellt sich als eine geeignete Wahl zur Beschreibung des Grades der Knäulung heraus. In einem Kurs mit 6 Gruppen wurden folgende Ergebnisse erhalten (vgl. Abb. D4):

$$\langle R^2 \rangle = \ 11{,}1 \mid 10{,}1 \mid 10{,}1 \mid 9{,}7 \mid 5{,}83 \mid 16{,}1$$

Anhand dieser Ergebnisse kann man bereits einen Mittelwert um 10 vermuten. Wenn dieser günstige Fall eintritt, können die Lernenden selbstständig zu der Gleichung $\langle R^2 \rangle = N$ gelangen. Der Zusammenhang dieser Gleichung mit der Gestalt der Polymere liefert der Vergleich mit gestreckten Molekülen wie in Abb. D2 erläutert, für die $\langle R^2 \rangle = N^2$ gelten würde. In den Simulationen wird aber stattdessen $\langle R^2 \rangle \approx 10$ erhalten d.h. die Moleküle müssen geknäult sein.



Tetraederwürfel: 1: rechts, 2: links, 3: hoch, 4: runter

Der Startpunkt bei (0|0) ist mit einem Kreis markiert.

Zahl der Segmente $N = 10$, d.h. für 10 Züge die Richtung würfeln

Nummer der Konfiguration (k)	Δx	Δy	R^2
1	6	-4	50
2	-3	-1	10
3	-5	5	50
4	0	0	0
5	4	0	16
6	0	0	0
7	-2	-4	20
8	-2	0	4
9	2	-2	8
10	1	1	1
Erwartungwert	$\langle \Delta x \rangle =$ 0,1	$\langle \Delta y \rangle =$ -0,7	$\langle R^2 \rangle =$ 15,9
	$\langle R \rangle = \sqrt{\langle \Delta x \rangle^2 + \langle \Delta y \rangle^2} =$		

Tabelle I

Gruppen	Δx	Δy	R^2
I	-0,4	-0,5	11,1
II	-0,2	-0,13	10,1
III	-0,9	-0,6	10,1
IV	0,7	0,2	9,7
V	0	-0,83	5,83
VI	0,1	-0,7	15,9
~~VII~~			
~~VIII~~			
Erwartungwert	$\langle \Delta x \rangle \approx -0{,}12$	$\langle \Delta y \rangle \approx -0{,}43$	$\langle R^2 \rangle \approx 10$
	$\langle R \rangle = \sqrt{\langle \Delta x \rangle^2 + \langle \Delta y \rangle^2} =$		$R^2 = N$

Tabelle II

Mittelwert von R^2: $\langle R^2 \rangle = N$

Abb. D4: Protokoll für die Durchführung der haptischen Simulation zur Gestalt eines Polymers mit flexiblen Bindungen.

[D1] Programmiersprache Scratch, https://scratch.mit.edu/ (letzter Zugriff 06.09.2023)

[D2] Kraska, T. (2021). CHEMKON, 28, 299–304.

E) Grundlegendes zu Scratch

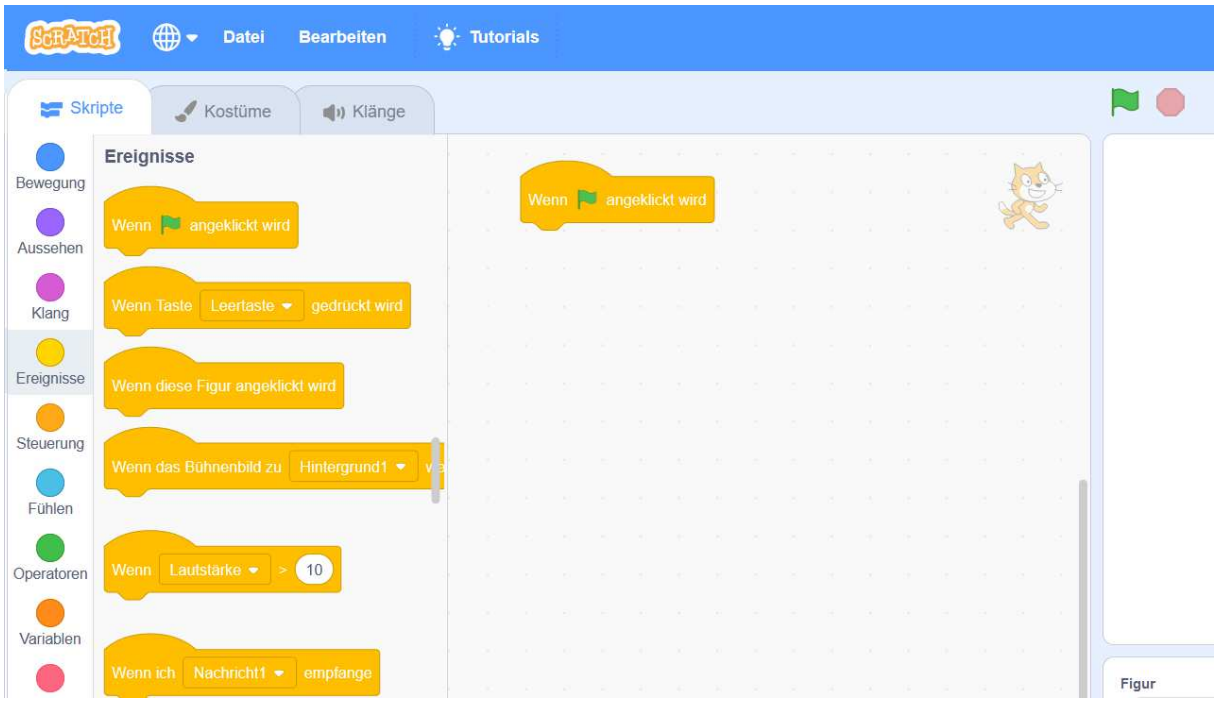
Scratch kann man auch ohne Installation eines lokalen Programms als Web-Anwendung laufen lassen, wenn man https://scratch.mit.edu/ aufruft und auf

✨ **Beginne mit dem Erstellen**

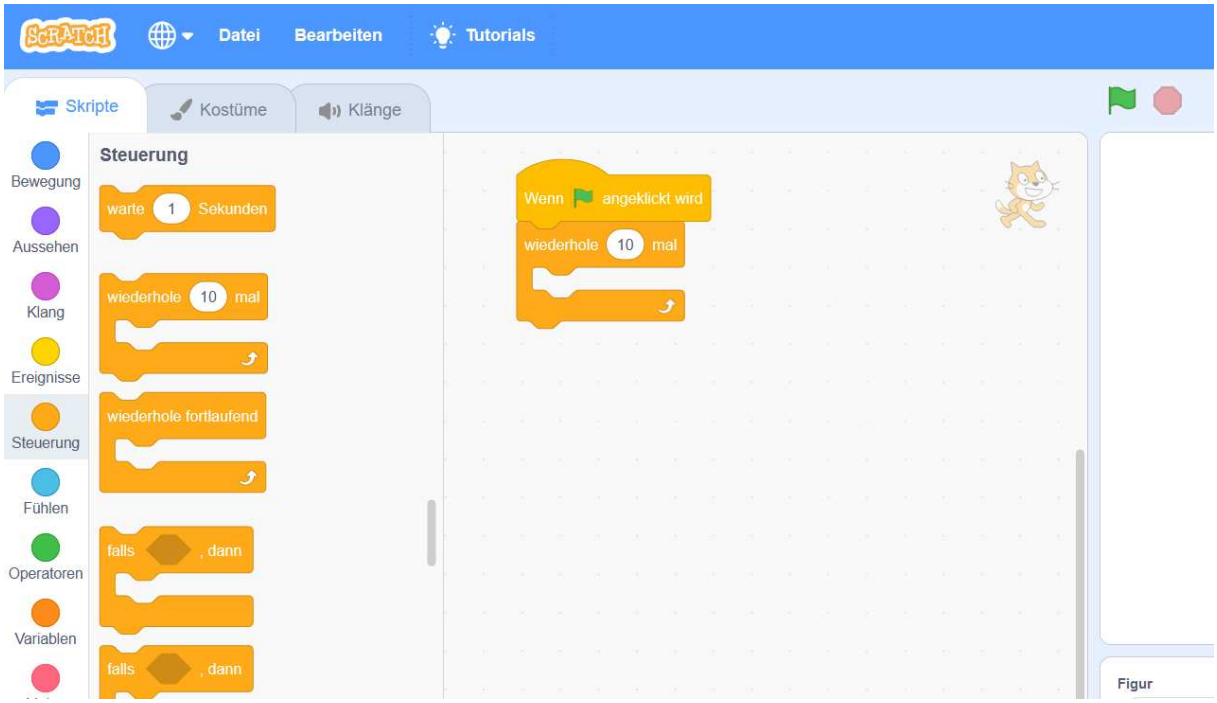
klickt. Es folgt ein einführendes Video nach dem man mit der Programmierung starten kann.

Die Befehle von Scratch verbergen sich hinter den einzelnen Punkten unter dem Menüpunkt „Skripte“. Für die Programme hier muss noch die Bibliothek „Malstift“ hinzugeladen werden. Dazu klickt man links unten auf das Symbol  (ggfs. runterscrollen) und klickt dann auf den Malstift.

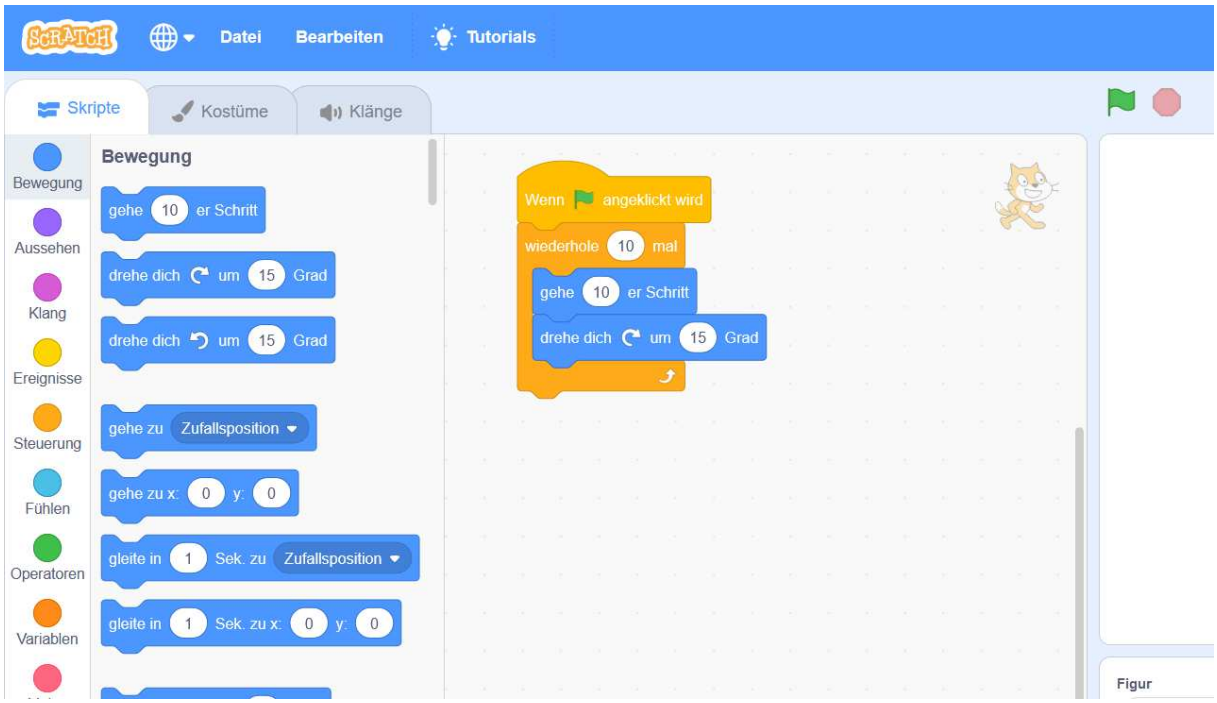
Im Folgenden ist die Erstellung eines Scratch-Programms für ein einfaches Beispiel dargestellt. Zunächst wird aus „Ereignisse“ der Start-Befehl in das Programmfeld mit der linken Maustaste gezogen.



Dann wird aus „Steuerung“ eine Schleife an den Start-Befehl angehängt. Den Wert „10“ kann man verändern.

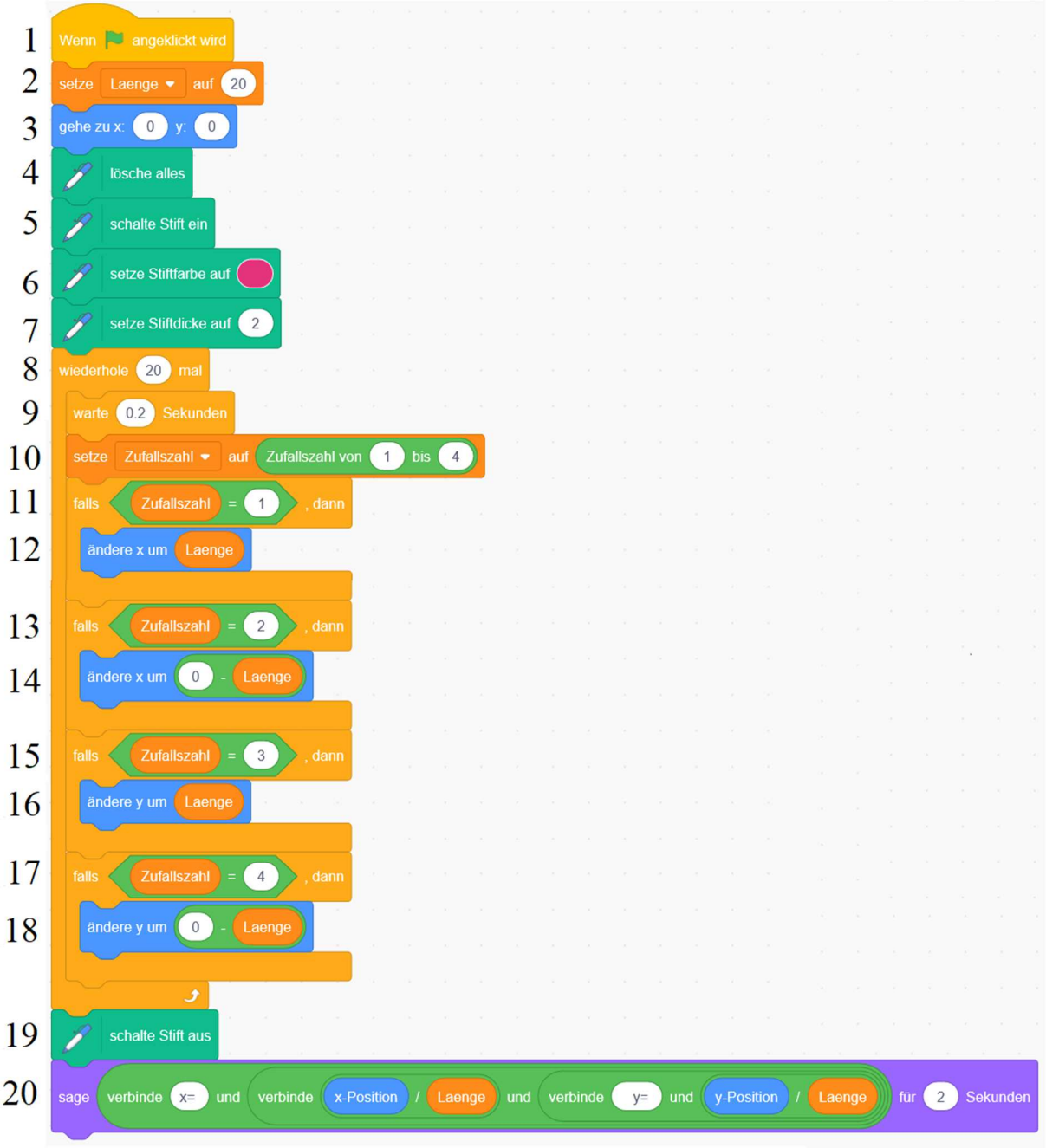


Aus „Bewegung“ wird der Befehl „gehe“ sowie „drehe“ in die Schleife hineingezogen. Mit Klick auf die grüne Fahne wird das Programm gestartet.



F) Simulation des Random Walks mit Scratch

In dem abgebildeten Programm wird ein Random Walk (RW) simuliert und am Ende in einer Sprechblase die Verschiebung in x- und in y-Richtung relativ zum Startpunkt ausgegeben. Dabei wird die Verschiebung durch die `Laenge` in Bildschirmpunkten geteilt, um die Einheitslänge für die Schrittweite zu erhalten. Außerdem sind auf der folgenden Seite Erläuterungen zum Code (P6) ergänzt.



Erläuterungen zum Code P6

1 Start durch Klick auf die Fahne

2 Definition der Variable `Laenge`. Der Wert wird auf 20 gesetzt. Dies ist die Zahl der Bildschirmpunkte für einen Schritt des RW.

3 Malstift auf den Ursprung setzen

4 Bildschirm löschen

5 Stift absetzen

6 Stiftfarbe auf rot setzen

7 Stiftdicke auf 2 setzen

8 Anfang der Schleife zur Erzeugung eines RWs mit 10 Schritten

9 eine kleine Pause von 0,2 Sekunden zur Verlangsamung der Simulation

10 erzeuge eine Zufallszahl zwischen 1 und 4

11 wenn die Zufallszahl 1 ist, dann

12 erhöhe die x-Koordinate um den Wert der Variablen `Laenge`

13 wenn die Zufallszahl 2 ist, dann

14 verringere die x-Koordinate um den Wert der Variablen `Laenge`

15 wenn die Zufallszahl 3 ist, dann

16 erhöhe die y-Koordinate um den Wert der Variablen `Laenge`

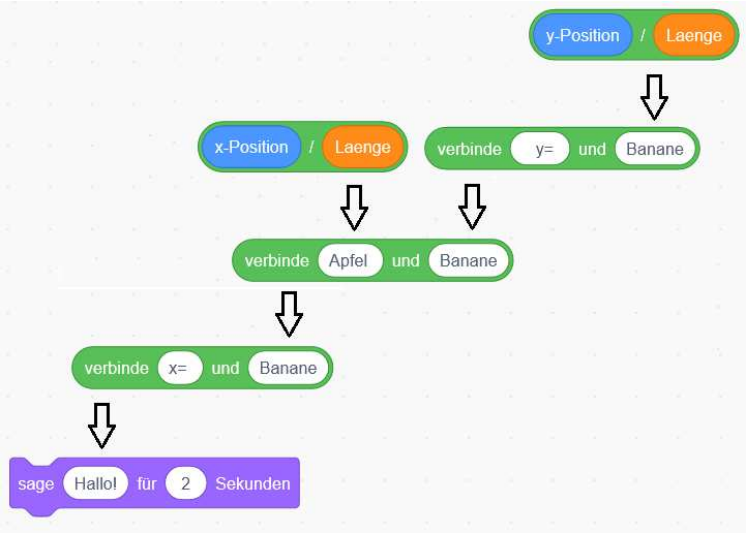
17 wenn die Zufallszahl 4 ist, dann

18 verringere die y-Koordinate um den Wert der Variablen `Laenge`

19 Stift wieder anheben

20 Bildschirmausgabe der Zahl der Schritte in x- und in y-Richtung

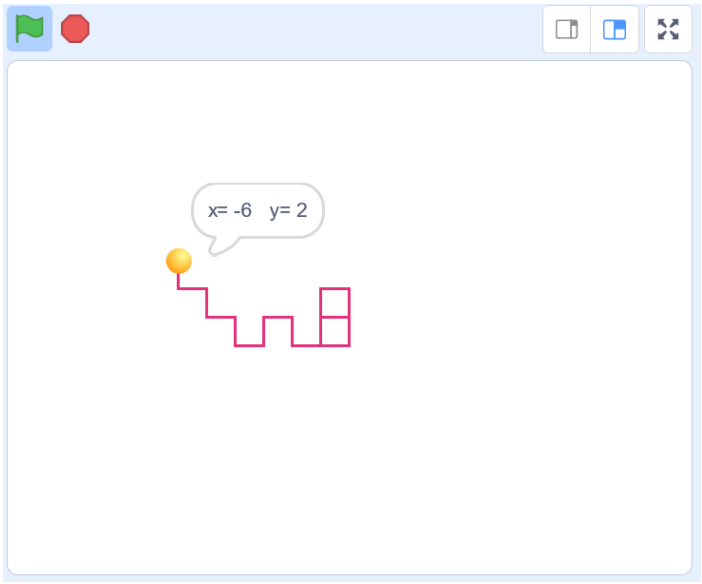
Der letzte Befehl ist aus mehreren Blöcken aufgebaut wie rechts abgebildet:



Der gleiche Befehl ist in einem Textcode wie TigeJython deutlich kompakter:

```
Print "x= ",xPosition/Laenge," y= ",yPosition/Laenge
```

Unten ist eine Ausgabe des Programms abgebildet.



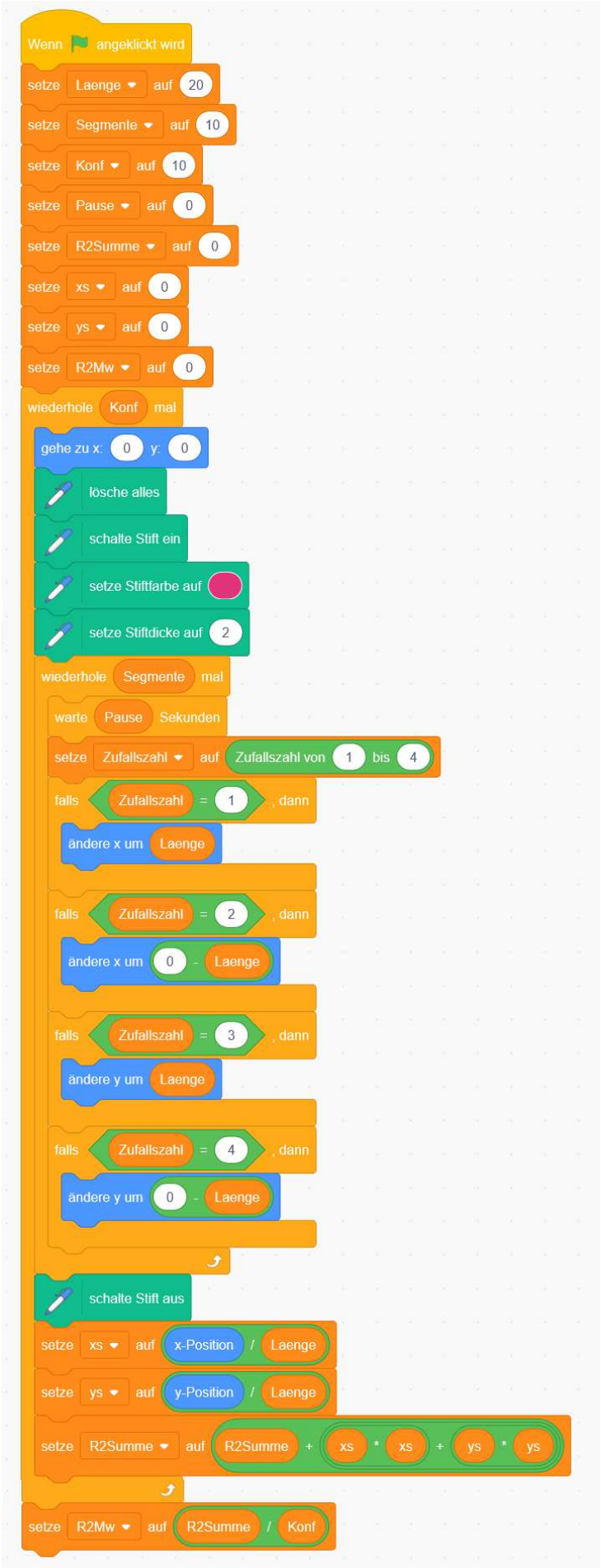
Berechnung des Mittelwerts des quadratischen End-End-Abstand $\langle R^2 \rangle$

Das folgende Programm ist eine Erweiterung des obigen, bei dem das Quadrats des End-End-Abstandes berechnet und über eine gegebene Anzahl von Konfigurationen gemittelt wird. Es handelt sich um Programm P7 (s. Kapitel A). Dazu muss eine Reihe von Variablen definiert werden und mit dem Befehl `setze` auf einen Wert gesetzt werden. Der Befehl `setze` hat die Funktion einer Zuweisung in Scratch wie das Gleichzeichen in textorientierten Programmiersprachen.

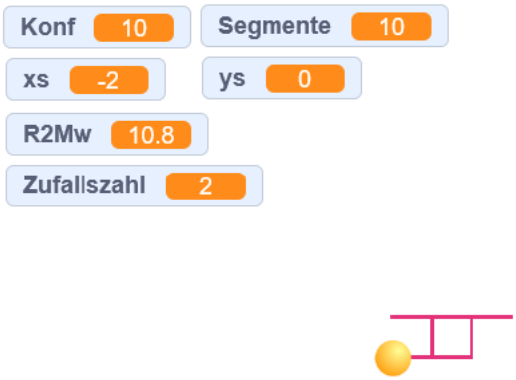
Die Berechnung des Radom Walks wird in einer Schleife ausgeführt und am Ende wird der Mittelwert $\langle R^2 \rangle$ (`R2Mw`) berechnet. Die Bildschirm-Position am Ende eines Random Walks wird durch die Länge eines Schritts (in Bildschirmpunkten) geteilt, um die Einheitslänge für den Schritt bei der Auswertung zu erhalten. Dazu werden die Variablen `xs` und `ys` definiert:

```
xs=xPosition/Laenge
```

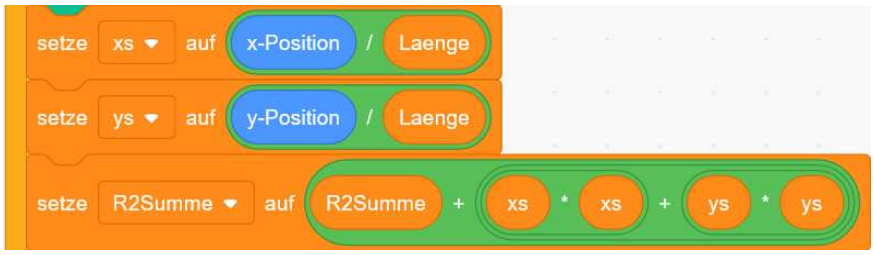
```
ys=yPosition/Laenge
```


Wenn angeklickt wird
setze Laenge auf 20
setze Segmente auf 10
setze Konf auf 10
setze Pause auf 0
setze R2Summe auf 0
setze xs auf 0
setze ys auf 0
setze R2Mw auf 0
wiederhole Konf mal
gehe zu x: 0 y: 0
lösche alles
schalte Stift ein
setze Stiftfarbe auf
setze Stiftdicke auf 2
wiederhole Segmente mal
warte Pause Sekunden
setze Zufallszahl auf Zufallszahl von 1 bis 4
falls Zufallszahl = 1 , dann
ändere x um Laenge
falls Zufallszahl = 2 , dann
ändere x um 0 - Laenge
falls Zufallszahl = 3 , dann
ändere y um Laenge
falls Zufallszahl = 4 , dann
ändere y um 0 - Laenge
schalte Stift aus
setze xs auf x-Position / Laenge
setze ys auf y-Position / Laenge
setze R2Summe auf R2Summe + xs * xs + ys * ys
setze R2Mw auf R2Summe / Konf

Unten ist ein Beispiel für eine Ausgabe abgebildet. Dies ist der Mittelwert für 10 Konfigurationen mit 10 Segmenten. Das Ergebnis ist $\langle R^2 \rangle = 10{,}8$.



Zwar ist Scratch intuitiv und kann auch ohne große informatische Vorbildung eingesetzt werden, allerdings ist es in der Praxis eher für kleine Probleme geeignet. Dies liegt an den eingeschränkten Möglichkeiten für Berechnungen aber auch die unzureichende Ausgabe und Übersicht über den Code auf dem Bildschirm. Unten ist die Addition des Abstandsquadrats zu der Variablen `R2Summe` in Scratch gelistet. Der darunter gelistete Code der textorientierten Programmiersprache TygerJython ist deutlich übersichtlicher und auch einfacher einzugeben. Für den Einsatz von Scratch sind demnach kleine Aufgaben gut geeignet, sobald das Programm rechenintensiv und umfangreicher wird, sind textorientierte Sprachen geeigneter.

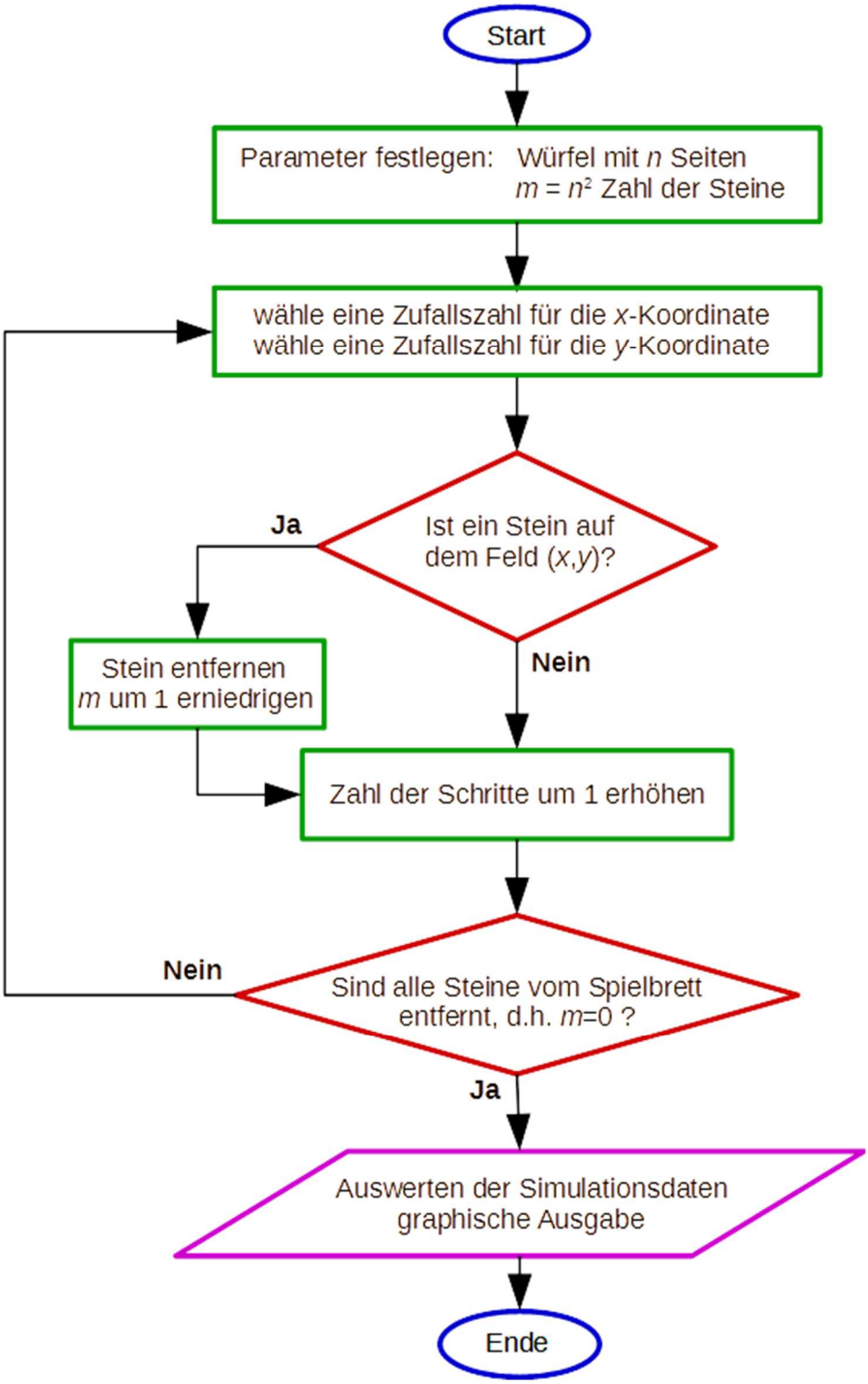


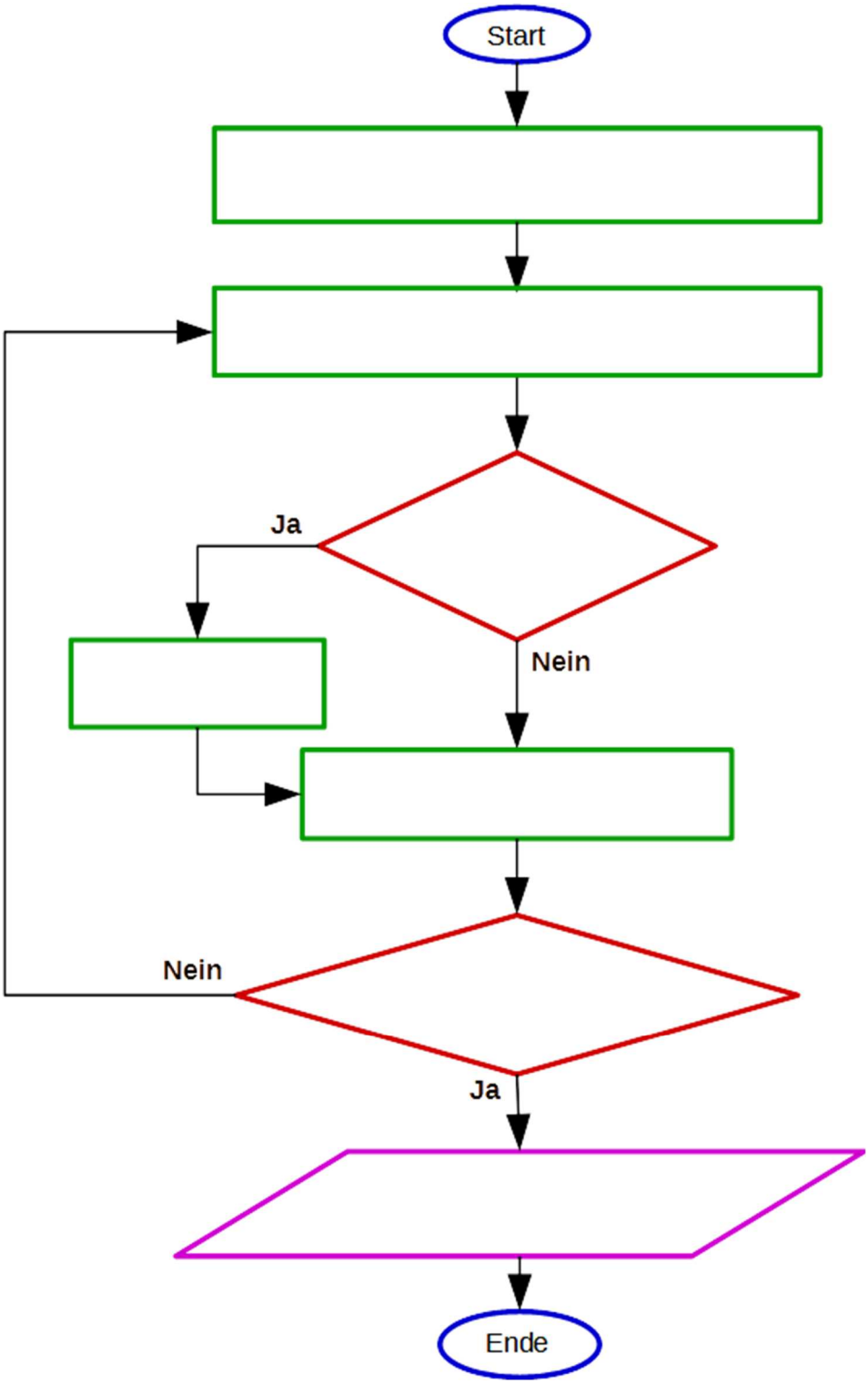
Der entsprechende TigerJython-Code lautet:

```
xs = xPosition / Laenge
ys = yPosition / Laenge
R2Summe = R2Summe + xs * xs + ys * ys
```

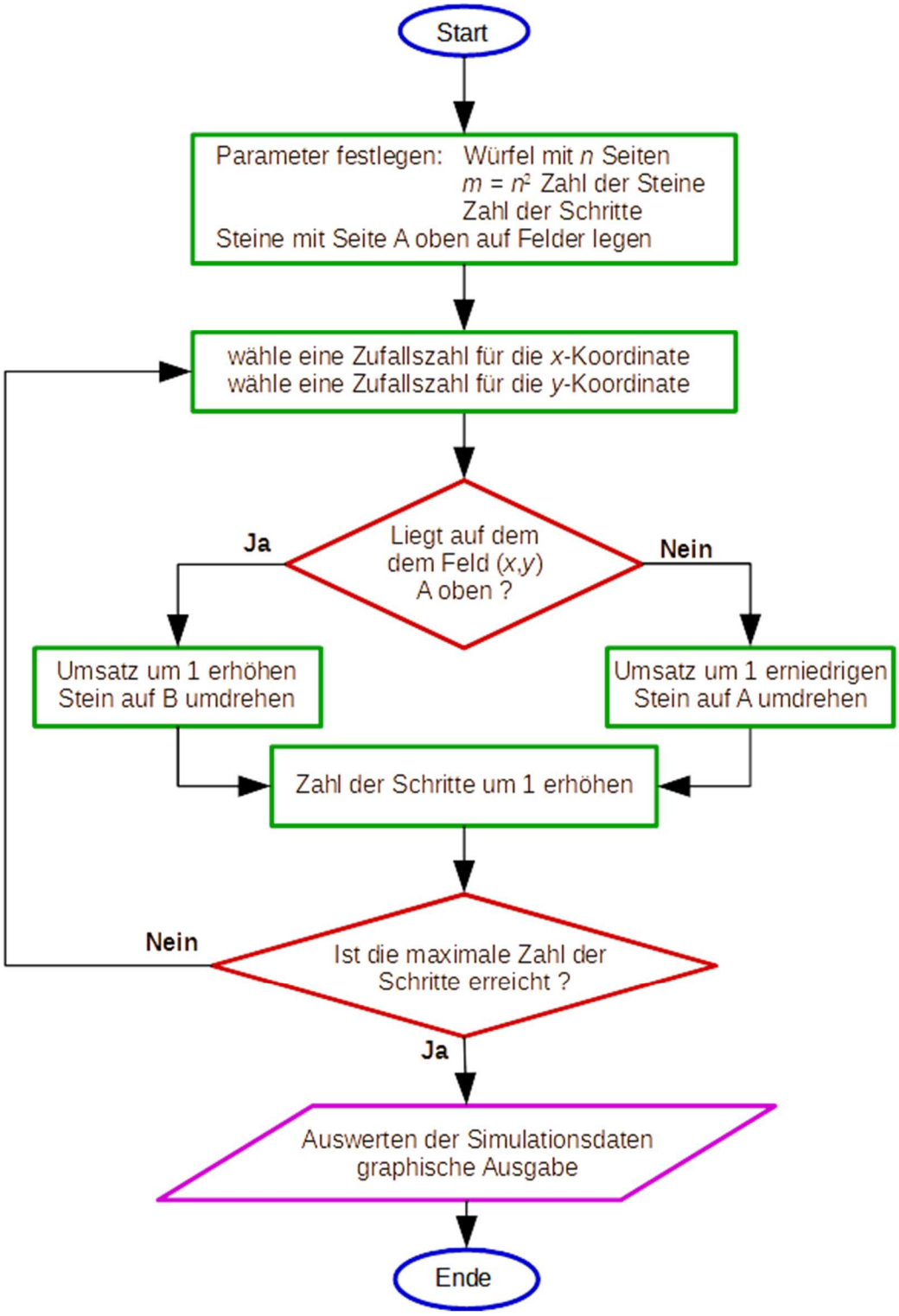
G) Programmablaufpläne

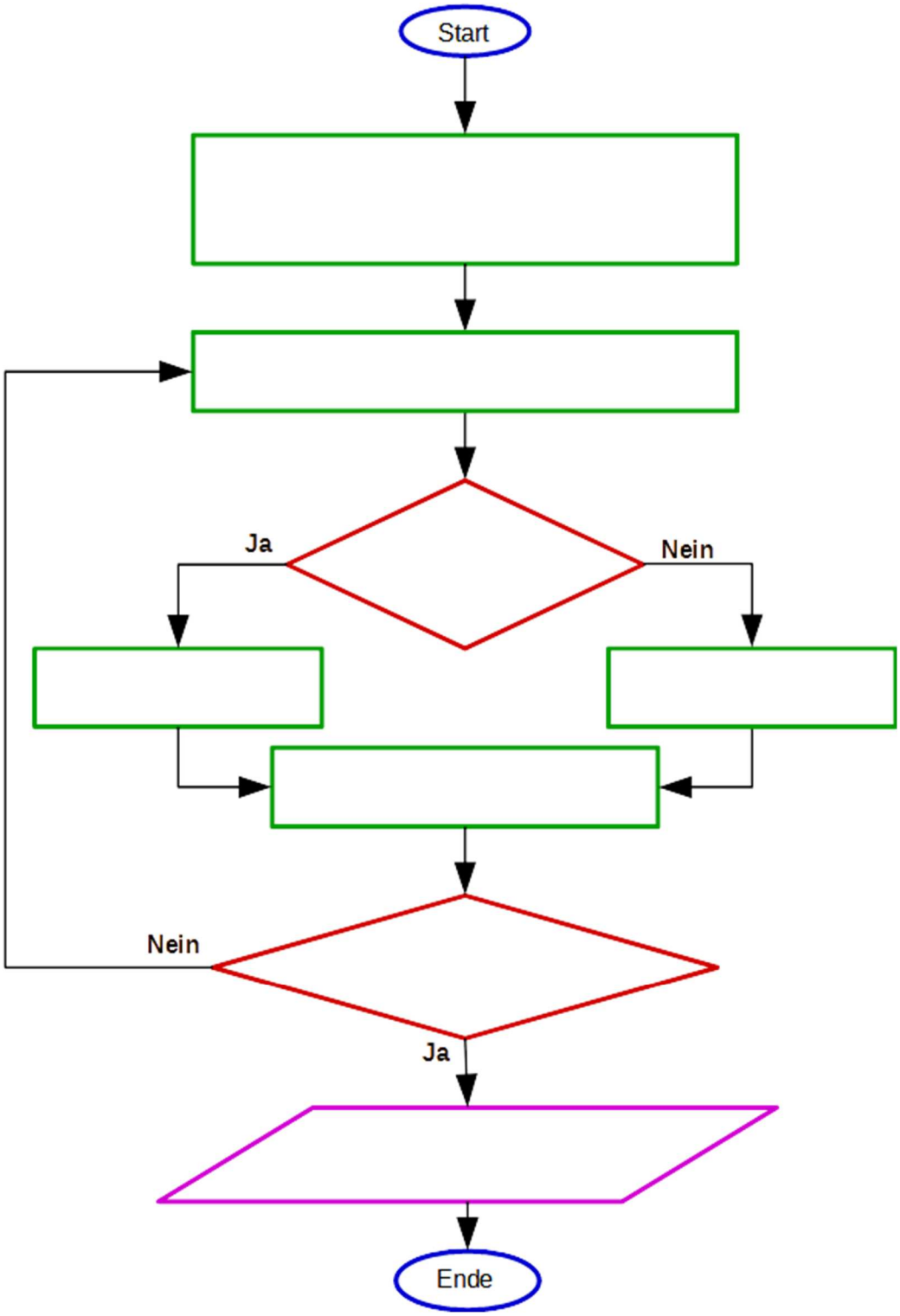
Programmablaufplan für die Simulation der **chemischen Kinetik** einer irreversiblen Reaktion 1. Ordnung.



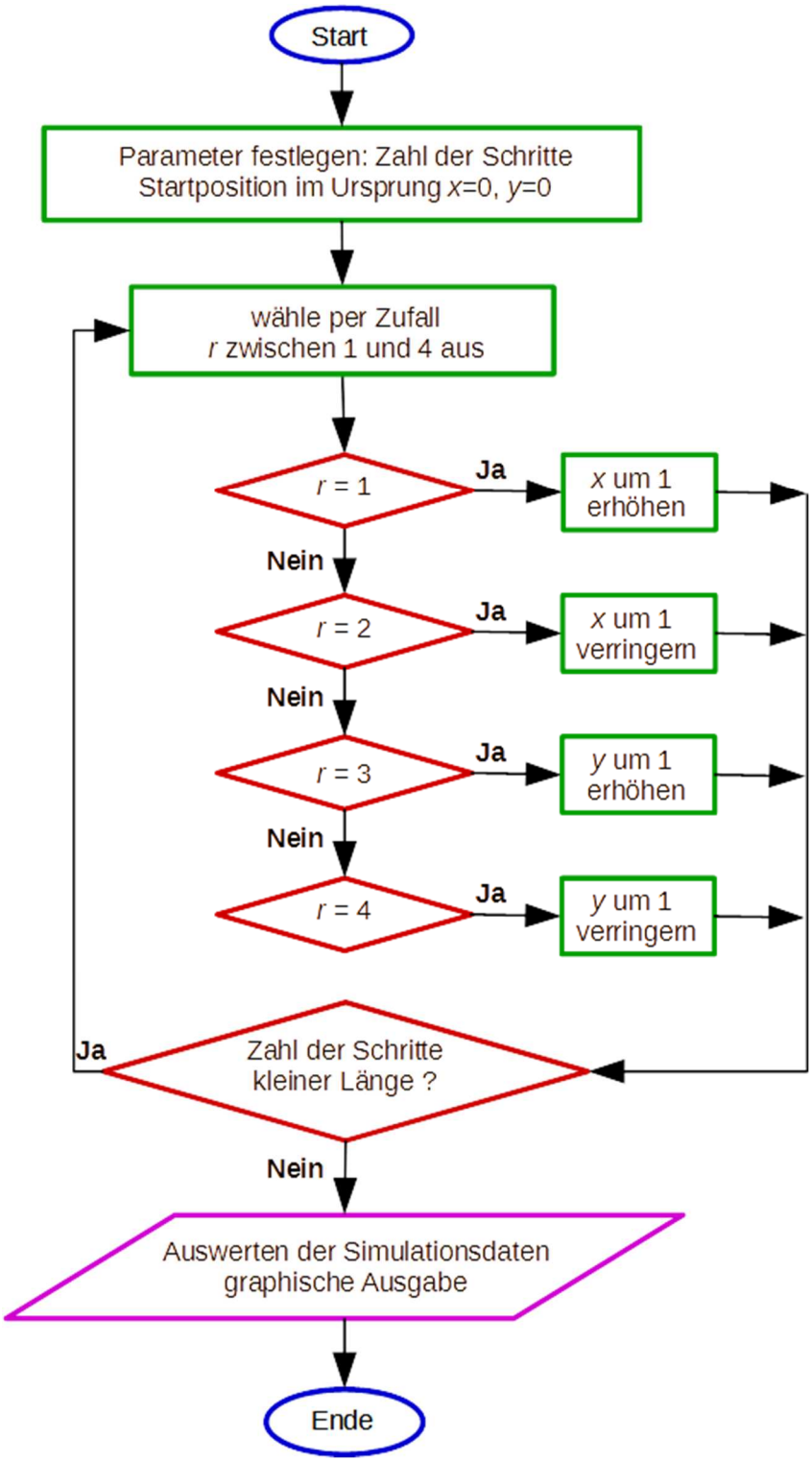

Start
Ja
Nein
Nein
Ja
Ende

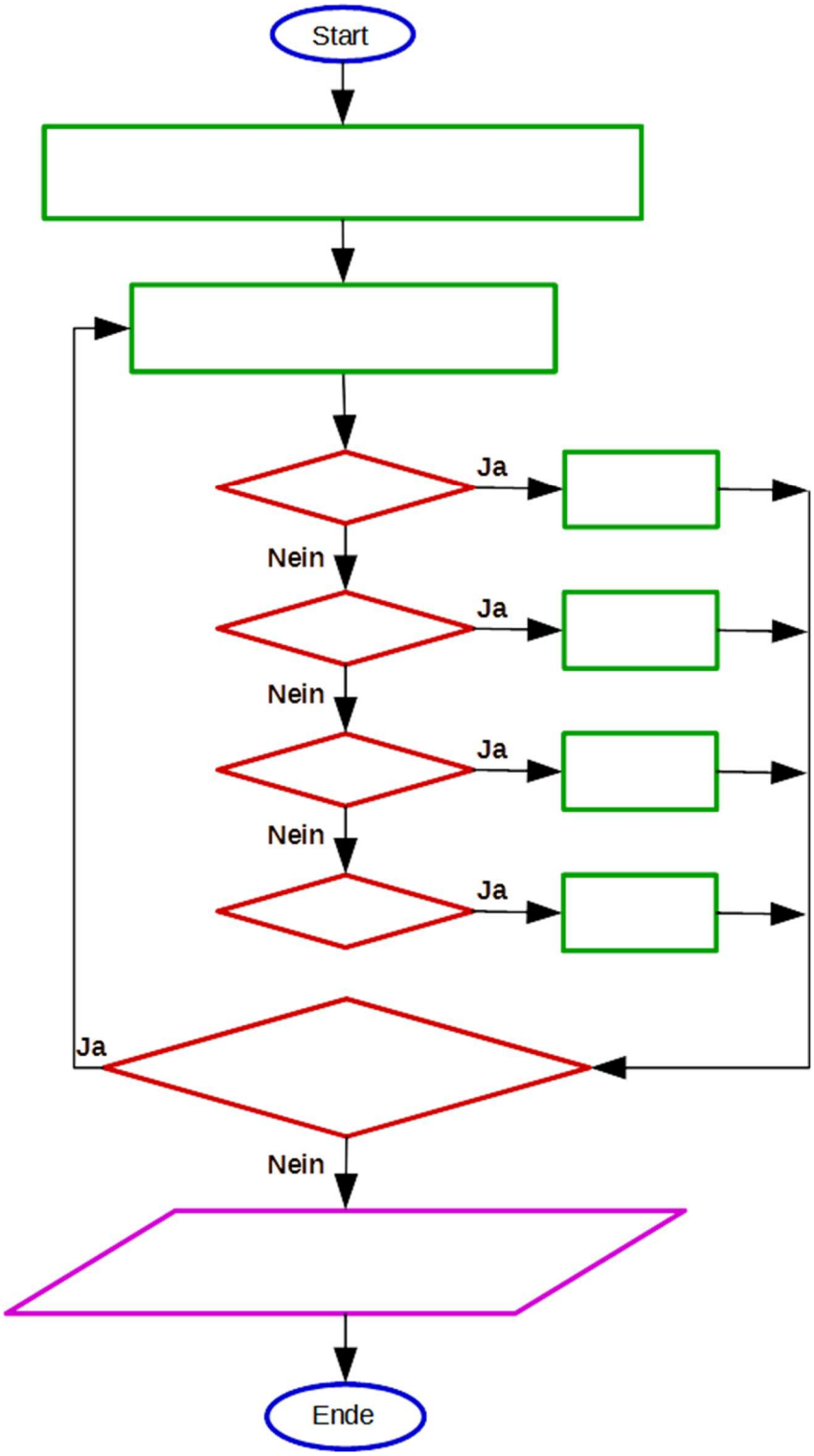
Programmablaufplan für die Simulation eines **chemischen Gleichgewichts**.




Start
Ja
Nein
Nein
Ja
Ende

Programmablaufplan für die Simulation eines **Random Walks** auf einem quadratischen Gitter zur Untersuchung der Struktur von Polymermolekülen.




Start
Ja
Nein
Ja
Nein
Ja
Nein
Ja
Ja
Nein
Ende