



„Die Kuhnschen Knäuel sind uns hier ein Gräuel“

Paradigmenwechsel durch Teilchensimulation im Chemieunterricht

THOMAS KRASKA

Online-Ergänzung

Online Ergänzungen zu dem Artikel:

„Die Kunhschen Knäuel sind uns hier ein Gräuel“ – Paradigmenwechsel durch Teilchensimulation im Chemieunterricht

Thomas Kraska

In den Online Ergänzungen sind vier Programmcodes zur Verfügung gestellt. Sie sind für die Entwicklungsumgebung TigerJython geschrieben. Diese kann man kostenlos herunterladen bei <https://tigerjython.ch/de/products/download>.

`P1-RW-Gitter.py`

Dies ist das in Abb. 1 abgebildete Programm.

`P2-RW-Gitter-Graphik.py`

Dies ist das gleiche Programm wie P1 zusätzlich mit Graphik-Ausgabe. Die Konfigurationen in Abb. 2 wurden mit diesem Programm erstellt.

`P3-RW-Turtle.py`

Dies ist das in Abb. 4 abgebildete Programm.

`P4-RW-3Turtle.py`

Dies ist einer Erweiterung des Programms P3 für drei Turtles und anschließender Berechnung von $\langle R^2 \rangle$ für die drei erzeugten Konfigurationen.

Auf den folgenden Seiten sind die Arbeitsblätter für den Unterricht zusammengestellt. Die Spielanleitung kann man laminieren und den Random-Walk mit einem abwischbaren Filzstift einzeichnen lassen. Auf der letzten Seite ist eine Bastelvorlage für einen Tetraederwürfel abgebildet.

1) Spielanleitung

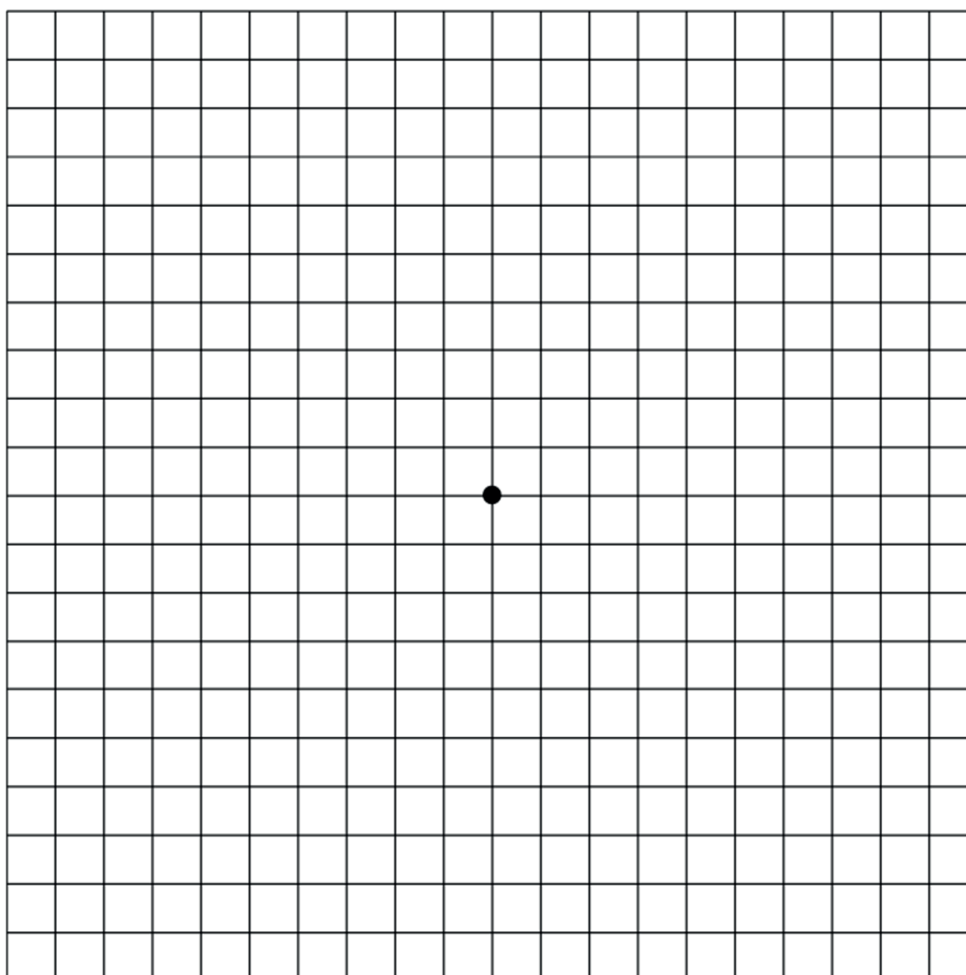
In diesem Spiel sollen Anordnungen (Konfigurationen) von Kettenmolekülen mit frei beweglichen Bindungen erzeugt werden, deren Ausrichtung zufällig ist. Der Anfang des Moleküls ist der schwarze Punkt auf dem Kästchenpapier.



Werfe 10 mal den Tetraederwürfel für Moleküle mit 10 Segmenten ($N = 10$). Ziehe einen Strich eine Einheit in die Richtung, die der gewürfelten Zahl entspricht (siehe unten).

Lesen anschließend den End-End-Abstand in beide Richtungen ab, also Δx und Δy , und trage sie in die Tabelle ein. Erzeuge so $k = 10$ Konfigurationen.

1 → 2 ← 3 ↑ 4 ↓



2) Auswertung der Ergebnisse

- Mittle die Δx und die Δy Werte über alle $k = 10$ Konfigurationen:
 $\langle \Delta x \rangle = \frac{1}{k} (\Delta x_1 + \Delta x_2 + \dots + \Delta x_k)$ $\langle \Delta y \rangle = \frac{1}{k} (\Delta y_1 + \Delta y_2 + \dots + \Delta y_k)$
- Berechne das Quadrat des End-End-Abstands $R^2 = \Delta x^2 + \Delta y^2$. Mittle dann über alle $k = 10$ Konfigurationen: $\langle R^2 \rangle = \frac{1}{k} (R_1^2 + R_2^2 + \dots + R_k^2)$

Nummer der Konfiguration	Δx	Δy	$R^2 = \Delta x^2 + \Delta y^2$
1			
2			
3			
4			
5			
6			
7			
8			
9			
10			
Mittelwerte	$\langle \Delta x \rangle =$	$\langle \Delta y \rangle =$	$\langle R^2 \rangle =$

- Berechne die Mittelwerte über alle Gruppen im Kurs:

Gruppe	$\langle \Delta x \rangle$	$\langle \Delta y \rangle$	$\langle R^2 \rangle$
I			
II			
III			
IV			
V			
VI			
VII			
VIII			
Mittelwerte	$\langle \Delta x \rangle =$	$\langle \Delta y \rangle =$	$\langle R^2 \rangle =$

- Diskutiere die Mittelwerte in der letzten Zeile und schlage einen Zusammenhang zwischen dem mittleren End-End-Abstand $\langle R^2 \rangle$ und der Zahl der Segmente N vor.

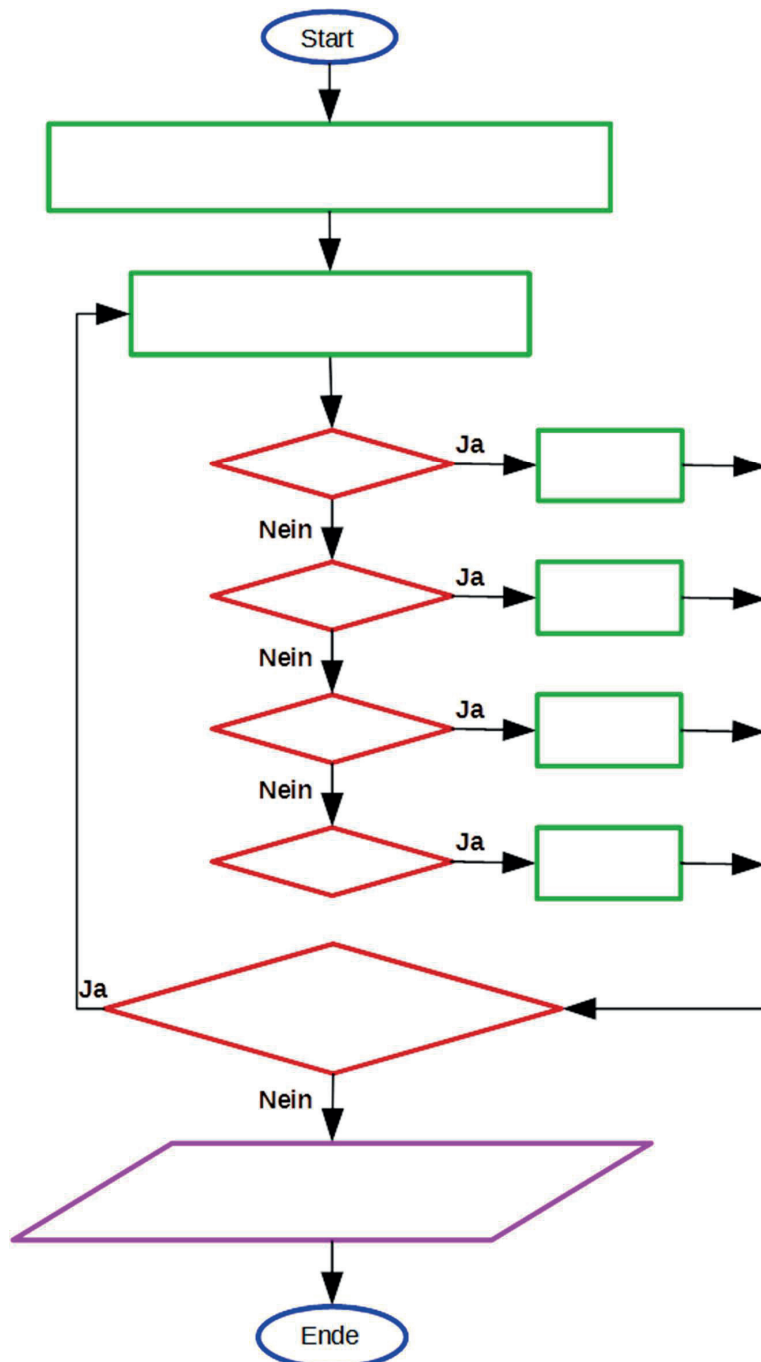
3) Pseudocode erstellen

Erstelle eine strukturierte Liste von detaillierten Anweisungen für den Ablauf des Spiels:

- _____
- _____
- _____
- _____
- _____
- _____
- _____
- _____
- _____
- _____
- _____
- _____
- _____
- _____
- _____
- _____
- _____
- _____

4) Programmablaufplan

Übertrage deine Anweisungsliste in den unten abgebildeten Programmablaufplan für die Simulation eines Random Walk auf einem quadratischen Gitter.



5) Programmcode

Gleiche den unten abgebildeten Programmcode mit der Liste von Anweisungen für das Spiel und dem Programmablaufplan ab. Ordne dazu die Befehlszeilen des Codes den Anweisungen und den Schritten im Programmablaufplan zu.

```

1 from random import randint
2 Konfigurationen = 1000
3 N = 20
4 rechts = 1
5 links = 2
6 hoch = 3
7 runter = 4
8 Summe = 0
9 for n in range(0, Konfigurationen):
10     x = 0
11     y = 0
12     for i in range(0, N):
13         Schritt = randint(1, 4)
14         if Schritt == rechts: x = x + 1
15         if Schritt == links: x = x - 1
16         if Schritt == hoch: y = y + 1
17         if Schritt == runter: y = y - 1
18     Summe += x**2 + y**2
19 Mittelwert = Summe / Konfigurationen
20 print "<R^2> =", Mittelwert, " N =", N

```

Einige Hinweise:

- Die eingerückten Programmteile hängen von der nicht eingerückten Zeile darüber ab. Eingerückt wird mit der TAB-Taste.
- x, y, N, rechts, ... sind Variablen, in denen je ein Wert gespeichert wird
- = ist zuweisend (die Variable links übernimmt den Wert von rechts)
- == ist ein vergleichendes Gleichzeichen
- += der Wert der Variablen auf der linken Seite wird um den Wert der rechten Seite erhöht; statt x=x+1 kann man auch x+=1 programmieren

