

Modul 411: Datenstrukturen und Algorithmen entwerfen und anwenden

Organisation von Modul 411 an der iet-gibb



**gibbix.ch → Die Lernumgebung der iet-gibb
bmLP1 (Ubuntu)**



Netz:



vmnet0: Bridged (DHCP)

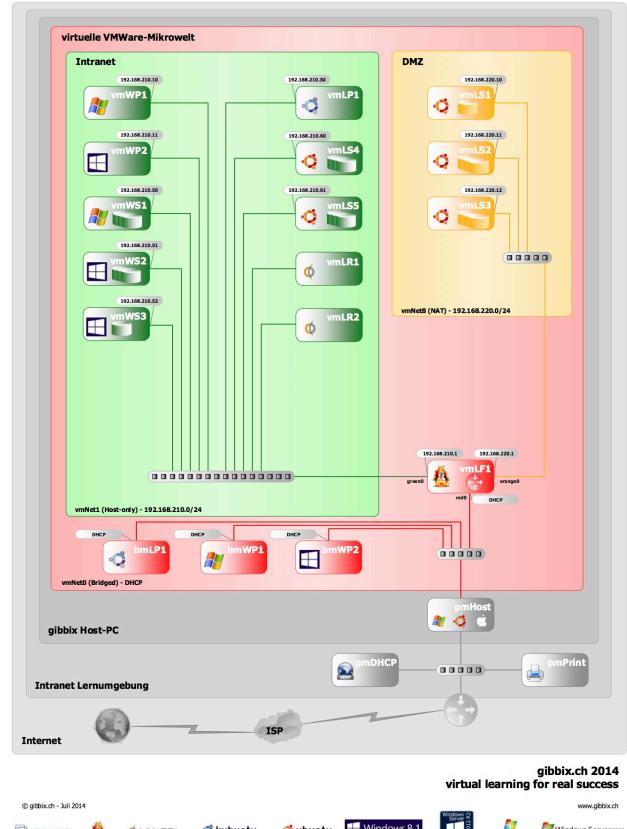
Programmierumgebung:
NetBeans IDE ist eine Entwicklungsumgebung, die für die Programmiersprache Java geschrieben wurde. Der Hersteller ist Oracle Corporation. Netbeans ist die offizielle IDE (Integrierte Entwicklungsumgebung) des Java Herstellers Oracle.

Modul 411 hat das Kompetenzfeld Application Engineering und setzt Erfahrungen im prozeduralen und objektbasierten Programmieren (Module 403 und 404) voraus.

Das Modul 411 beinhaltet folgende **Handlungsziele** festgelegt durch die ICT-Berufsbildung Schweiz:

1. Für ein gegebenes Problem eine geeignete Datenstruktur definieren und mit den Mitteln einer Programmiersprache, wie Structs, Referenzen / Zeiger und Arrays umsetzen.
2. Ein Problem analysieren und einen geeigneten Algorithmus zur Lösung mit den Grundelementen Zuweisung, Verzweigung und Schleife entwerfen und mit Prozeduren und Funktionen umsetzen.
3. Algorithmen und Datenstrukturen hinsichtlich Speicher- und Zeitkomplexität analysieren und dokumentieren.
4. Ein komplexeres Problem auf kleinere Teilprobleme zurückführen und je nach Problemstellung Iteration oder Rekursion einsetzen.

Lernarrangement „Three-Homed-Firewall“-Architektur



Virtual learning for real success <http://www.gibbix.ch>

5. Abstrakte Datentypen, wie Liste, Set, Map etc. und die darauf definierten Operationen kennen und zielgerichtet einsetzen können.
6. Datenstrukturen und Algorithmen mit dem Debugger und weiteren Tools untersuchen und dabei speziell die Situation auf Stack und Heap analysieren und in geeigneter Form darstellen.

Einführung

Algorithmus

Wenn wir Computerprogramme schreiben, versuchen wir in der Regel ein Verfahren zu finden, welches uns erlaubt, eine Vielzahl von Problemen zu lösen. Dieses Verfahren ist unabhängig von einer Programmiersprache und somit gleichermaßen geeignet für viele Betriebssysteme und Programmiersprachen. Es handelt sich um eine **schriftweise Methode, eine Abfolge von Anweisungen und Bedingungen**, um ein oder mehrere Probleme zu lösen. Der Begriff Algorithmus wird in der Informatik verwendet, um ein Problemlösungsverfahren zu beschreiben, dass in **wohldefinierten Einzelschritten** innerhalb eines Computerprogramms implementiert ist.

Algorithmen treffen wir in verschiedenen Bereichen des Lebens an. Zum Beispiel lernten wir in der Grundschule wie man auf Papier grosse Zahlen addiert, subtrahiert, multipliziert und dividiert.

Beispiel Addition:

$$\begin{array}{r} 141175 \\ + 4415 \\ \hline 145590 \end{array}$$

Anweisung 1:	$5 + 5 =$	10	(Behalte 1)
Anweisung 2:	$7 + 1 + 1 =$	9	
Anweisung 3:	$1 + 4 =$	5	
Anweisung 4:	$1 + 4 =$	5	
Anweisung 5:	$4 + 0 =$	4	
Anweisung 6:	$1 + 0 =$	1	

$$\rightarrow = 145590$$

Dieses einfache Beispiel aus der Grundschule zeigt, wie man schrittweise vorgeht, um grosse Zahlen zu addieren.

Herkunft des Wortes "Algorithmus"

- Muhammad ibn Musa abu Djafar al-Choresmi ca. 780-850 n. Chr.
- arabischer Mathematiker, geboren in Choresmien (heute Usbekistan)
- arbeitete in Bagdad im "Haus der Weisheit"
- war beteiligt an der Übersetzung der Werke griechischer Mathematiker ins Arabische
- schrieb ein "Kurzgefasstes Lehrbuch für die Berechnung durch Vergleich und Reduktion" (Lehrbuch zum Rechnen mit Dezimalzahlen)
- lateinische Übersetzung dieses Buches ("liber algorismi") kam durch Kreuzfahrer nach Europa
- verfasste auch ein Buch mit dem Titel "Al-Mukhtasar fi Hisab al-Jabr wa l-Muqabala"



Denkmal in Xiva (Khiva; kyrillisch Хива) ist eine Oasenstadt in Usbekistans



Eine Seite aus Al-Mukhtasar fi Hisab al-Jabr wa l-Muqabala



Muhammad ibn Musa abu Djafar al-Choresmi auf einer sowjetischen Briefmarke.

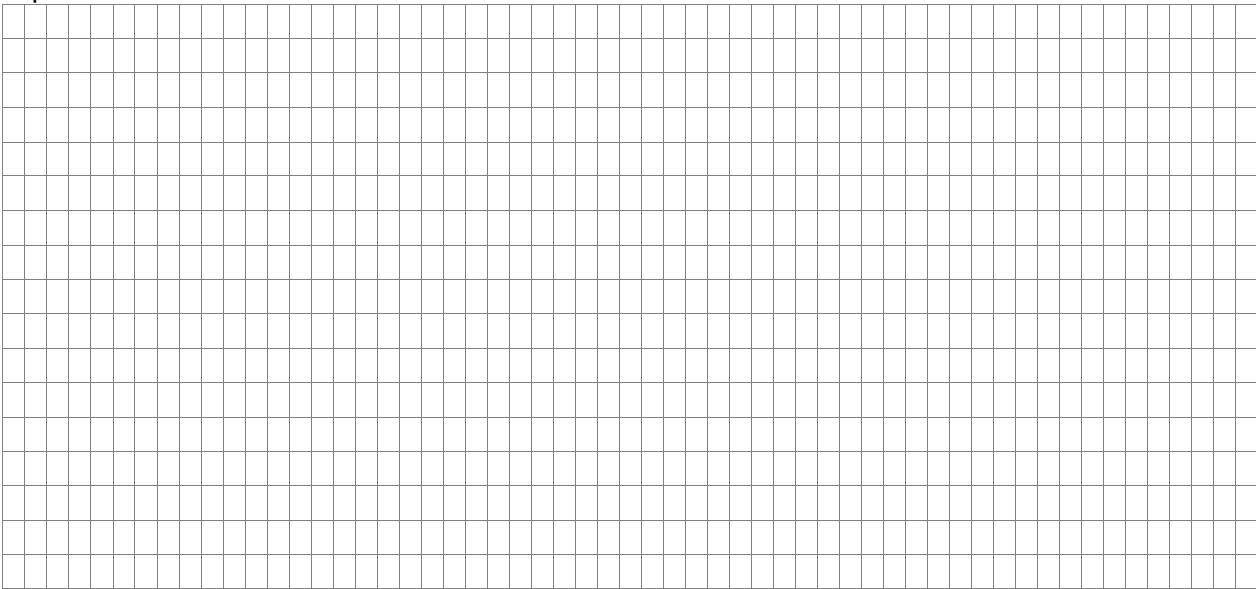
Aufgabe 1a:

Es gibt aber auch ganz andere Beispiele von Algorithmen, wie **Kochrezepte**. Hier das Beispiel einer Anleitung zum Backen von Muffins:

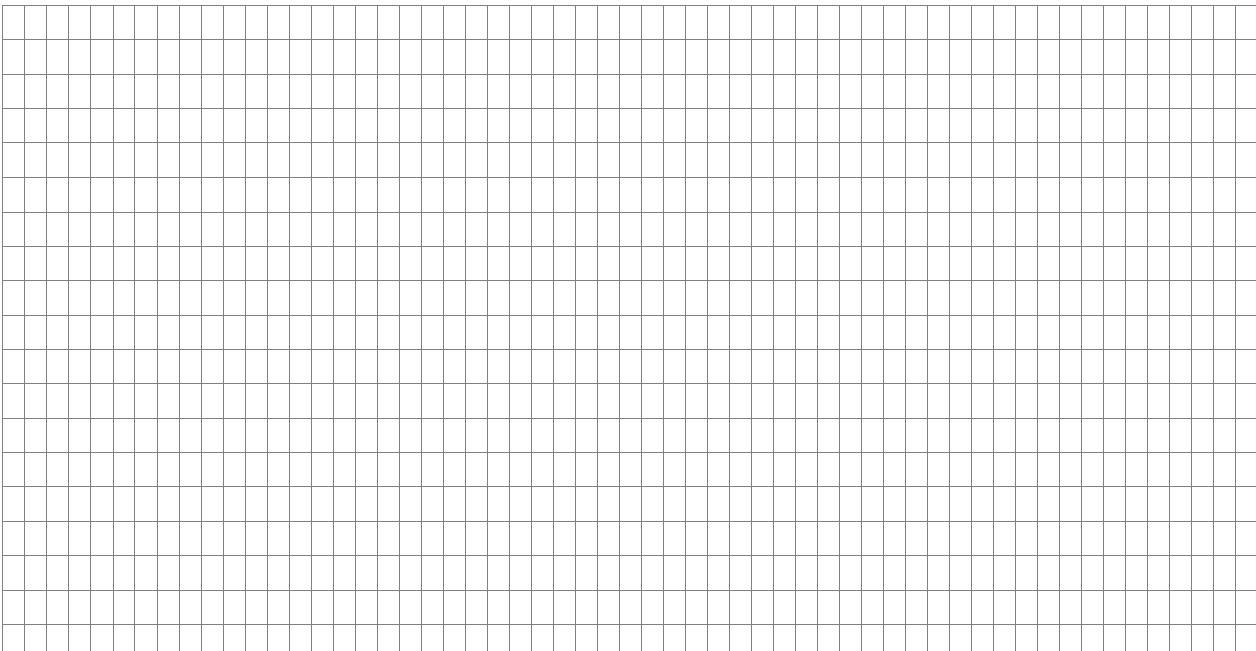
Muffin-Rezept:

Verrühre 200g Butter, 200g Zucker, 4 Eier, 1 Päckchen Backpulver, 1 Päckchen Vanillezucker, 250 g Mehl, 3 Esslöffel Rum und falls Apfelzeit - drei mittelgroße geschälte und zerteilte Äpfel; sonst füge 100g Schokoladenstückchen hinzu; fülle den Teig in Muffinförmchen; backe bei 175-200 Grad für etwa 30 min; bestäube die Muffins mit etwas Puderzucker.

Schreiben Sie das Kochrezept für Muffins in eine **Abfolge von Anweisungen**. Denken Sie daran, dass die Apfelzeit eine **bedingte Anweisung** ist, das heisst, dass die Bedingung Apfelzeit nicht immer erfüllt ist:

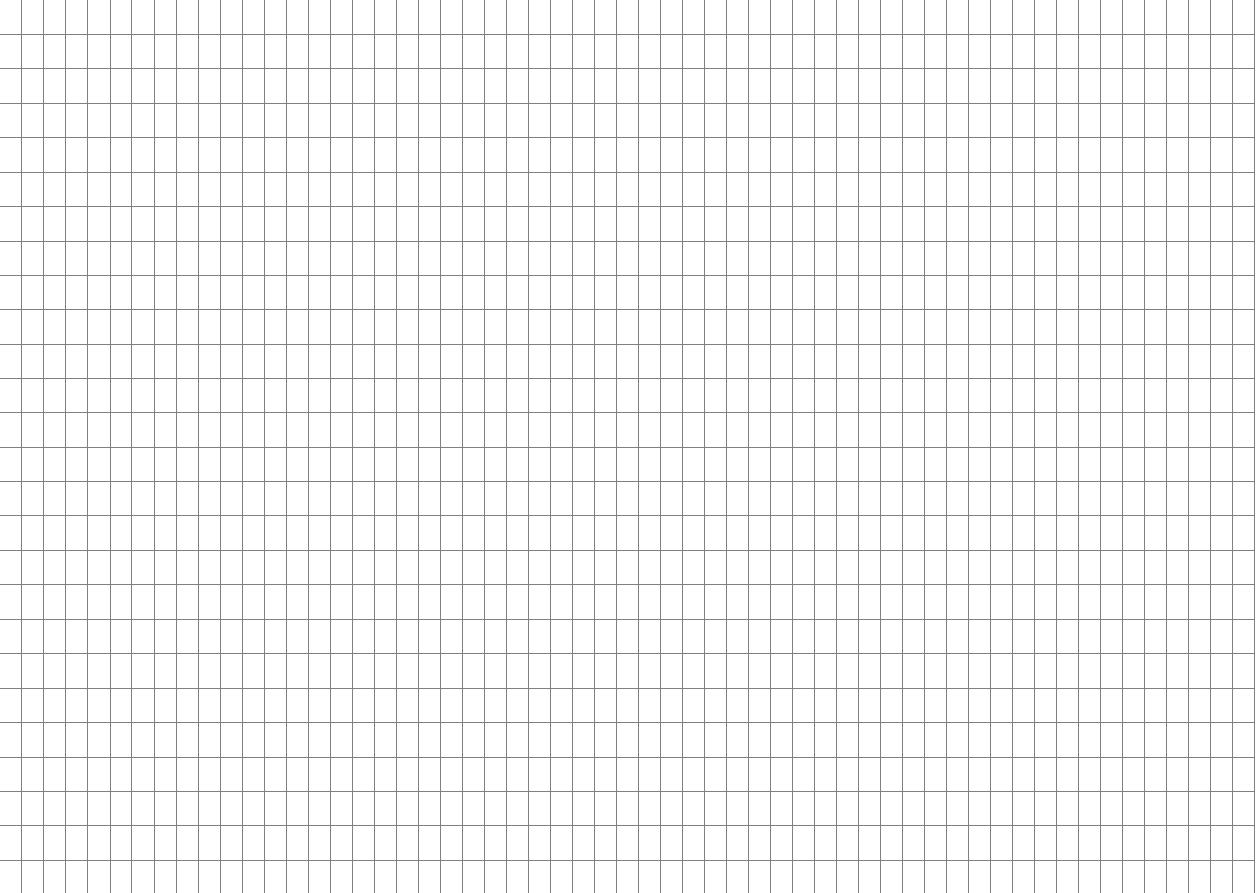
**Aufgabe 1b:****Muffin-Rezept unter Berücksichtigung der Backzeit:**

Implementieren Sie als Iteration (Schleife) die Backzeit für das Muffin-Rezept. Gehen Sie davon aus, dass Sie die Backzeit regelmässig überprüfen, um festzustellen, ob die Muffins gar sind:



Aufgabe 1c:**Muffin-Rezept für eine beliebige Anzahl Personen erweitern:**

Berücksichtigen Sie im Muffin-Kochrezept die Anzahl Personen. Parametrisieren Sie den Algorithmus durch eine Variable, die von aussen gesetzt werden kann (z. B. per Kommandozeile, aus einem File oder über eine Benutzeroberfläche kurz GUI). Die Anzahl Personen (Input) können vom Anwender frei gewählt werden.



Die drei Ebenen des Algorithmenentwurfs

1. Spezifikation

Vor dem Schreiben eines Programmes muss das Problem spezifiziert werden. Es stellen sich präzise Fragen, die von Vorteil vor der eigentlichen Programmierarbeit beantwortet werden.

2. Algorithmus

Vor dem Schreiben eines Programmes muss der genaue Ablauf von elementaren Aktionen, die das Problem schrittweise lösen, klar sein. Alle Einzelschritte und Vorbedingungen müssen bekannt sein und können mit zahlreichen Methoden dargestellt werden.

3. Programm

Konkrete Formulierung des Algorithmus in einer Programmiersprache (sog. "glue code"). In Modul 411 wenden wir Java an.

Im kommenden Abschnitt gehen wir genauer auf die zwei ersten Ebenen des Algorithmenentwurfs ein.

Spezifikation

Als Beispiel nehmen wir den euklidischen Algorithmus des grössten gemeinsamen Teiler $\text{ggT}(a, b)$. Für beliebige Zahlen a und b wird der $\text{ggT}(a, b)$ berechnet, also die grösste Zahl, die sowohl a und b teilt.

Wir unterscheiden in der Spezifikation zwischen Pre-conditions (Vorbedingungen) und Post-conditions (Nachbedingungen).

Beispiele von **Pre-conditions** für den $\text{ggT}(a, b)$:

1. Welche Zahlen a, b sind zugelassen?
→ Positive oder auch negative Zahlen?
2. Welche Grundoperationen sind erlaubt?
→ '+', '-' oder auch `div` und `mod`?

Beispiele für **Post-conditions** für den $\text{ggT}(a, b)$:

1. Was wird ausgegeben, falls $m, n < 0$?
2. Was wird ausgegeben, falls die Eingabe nicht den pre-conditions genügt?

Pre-conditions umfassen alle relevanten Eigenschaften, die vor Ausführung des Algorithmus gelten.

Post-conditions umfassen alle relevanten Eigenschaften, die nach Ausführung des Algorithmus gelten.

Das Praxisproblem der Spezifikation

In der Praxis werden häufig weniger formale Beschreibungen in natürlicher Sprache verwendet (Pflichtenhefte).

- Häufig umfangreich, mehrdeutig, inkonsistent
- Ist die Ausgabespezifikation das, was der Kunde wollte?
- Bekommt der Kunde von seinen "Zulieferern" das, was die Eingabespezifikation sagt?
- Tut der Algorithmus das, was in der Spezifikation steht?
- Tut der Algorithmus das, was der Kunde wollte?

Tipp: versuchen Sie, für Algorithmen immer eine "möglichst formale" Spezifikation zu schreiben!

Algorithmus

Definition von Algorithmus nach Adam Riese:

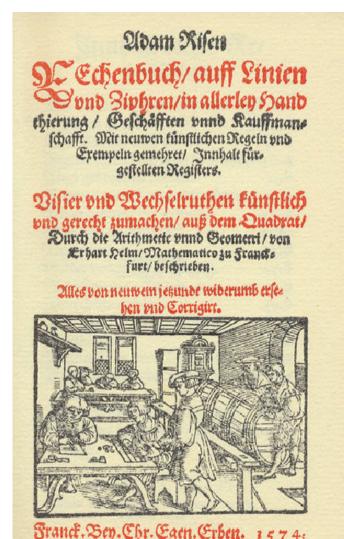
Geordnete Menge eindeutiger, diskreter, effektiv durchführbarer Schritte, die die gegebene Vorbedingung in endlich vielen Schritten in die gegebene Nachbedingung überführen.

Adam Riese schreibt von einer präzisen Vorschrift, wie in endlich vielen Schritten ein Problem gelöst werden kann!

Beispiel aus dem Rechenbuch von Adam Riese 1574:

Duplieren Lehret wie du ein zahl zweyfaltigen solt. Thu ihm also:
Schreib die zahl vor dich /mach ein Linien darunter/ heb an zu
forderst / Duplir die erste Figur. Kompt ein zahl die du mit einer
Figur schreiben magst / so setz die unden. Wo mit zweyey /
schreib die erste / Die ander bahalt im sinn. Darnach duplir die
ander / und gib darzu / das du behalten hast / und schreib
abermals die erste Figur / wo zwo vorhanden / und duplir fort biß
zur letzten / die schreibe gantz auß / als folgende Exempel
aufweisen.

aus: A. Risen, Rechenbuch, 1574



Begriffe in der Definition für "Algorithmus" nach Adam Riese

Ausführung des Algorithmus erfolgt in **diskreten Schritten**:

- einzelne, einfache Elementaraktionen
- Welche dies sind hängt vom sog. Algorithmischen Modell ab
 - Maschineninstruktionen?
 - Hochsprachliche Konstruktionen (JAVA)?
 - Mathematische Funktionen wie z.B. $\log(x)$?

Geordnete Menge:

Die Reihenfolge der Schritte genügt gewissen Regeln.

- Schritte müssen nicht unbedingt nacheinander ausgeführt werden.
- Es ist erlaubt, dass mehrere Schritte parallel von verschiedenen Recheneinheiten ausgeführt werden (parallele oder verteilte Algorithmen).
- Bei einer einzigen aktiven Recheneinheit allerdings gibt es einen ersten, zweiten, dritten Schritt usw. (sequentielle Algorithmen).

Eindeutigkeit:

Der Verfahrensablauf ist zu jedem Zeitpunkt determiniert, d.h.,

- zum Zeitpunkt der Ausführung eines Schrittes muss eindeutig und vollständig festgelegt sein, was zu tun ist; und
- gleiche Eingaben sollen zum gleichen Ablauf und Ergebnis führen.
- Ausnahme: randomisierte Algorithmen, deren Ablauf von (mathematisch bestimmten) Zufallsgrößen abhängt. Weitere Ausnahme ist die Parallelisierung durch Recheneinheiten.
-

Effektivität:

Jeder Schritt des Verfahrens muss effektiv mechanisch oder rechenbar ausführbar sein.

- Bemerkung: "Effektivität" (= erzielt das gewünschte Resultat) ist nicht zu verwechseln mit "Effizienz" (= Wirtschaftlichkeit)

Terminierung:

Der Algorithmus hält nach endlich vielen Schritten mit einem Ergebnis an. Sofern die Eingaben der Eingabespezifikation genügen.

Nicht erlaubt sind z.B. Schrittfolgen, die nie ein Ergebnis liefern:

1. Starte mit Zahl 0
2. Addiere 2
3. Falls Ergebnis ungerade, halte an, sonst weiter bei Schritt 2.

Korrektheit:

Korrektheit heisst, dass jedes berechnete Ergebnis der Ausgabespezifikation genügt, sofern die Eingaben der Eingabespezifikation eingehalten sind. D.h., Ein- und Ausgabeparameter genügen im Falle der Terminierung immer den Spezifikationen.

Es gibt zahlreiche Methoden zur Darstellung eines Algorithmus:

1. Pseudocode
2. Flussdiagramme
(Ablaufdiagramm)
3. Struktogramme
4. UML-Diagramme
5. Programmcode

Beispiel Pseudocode:

Finden des grössten Elements in Array

```
Algorithm maxOfArray( a, n ):
```

Input:

a = array of integers

n = Anzahl Elemente in a

Output:

```
max{ a[0], ..., a[n-1] }
```

```
Index m mit a[m] = max
```

```
currentMax = a[0]
```

```
indexOfMax = 0
```

```
for i = 1 .. n-1:
```

```
    if a[i] > currentMax:
```

```
        currentMax = a[i]
```

```
        indexOfMax = i
```

```
output currentMax, indexOfMax
```

Aufgabe 2:**Einleitung**

Das **Collatz-Problem** (auch als **ULAM Funktion** oder als **(3n+1)-Vermutung**) bezeichnet, ist ein **ungelöstes mathematisches Problem**, das 1937 von Lothar Collatz gestellt wurde.

Bei dem Problem geht es um Zahlenfolgen, die nach einem einfachen Bildungsgesetz konstruiert werden:

1. Beginne mit irgendeiner natürlichen Zahl $n > 0$.
2. Ist n gerade, so nimm als Nächstes $n/2$,
3. Ist n ungerade, so nimm als Nächstes $3n + 1$.
4. Wiederhole die Vorgehensweise mit der erhaltenen Zahl, solange $n \neq 1$.

So erhält man zum Beispiel für die Startzahl $n = 19$ die Folge

19, 58, 29, 88, 44, 22, 11, 34, 17, 52, 26, 13, 40, 20, 10, 5, 16, 8, 4, 2, 1

Anscheinend mündet jede Folge mit $n > 0$ in den Zyklus 4, 2, 1 unabhängig davon, welche Startzahl n man probiert hat. Die Collatz-Vermutung lautet:

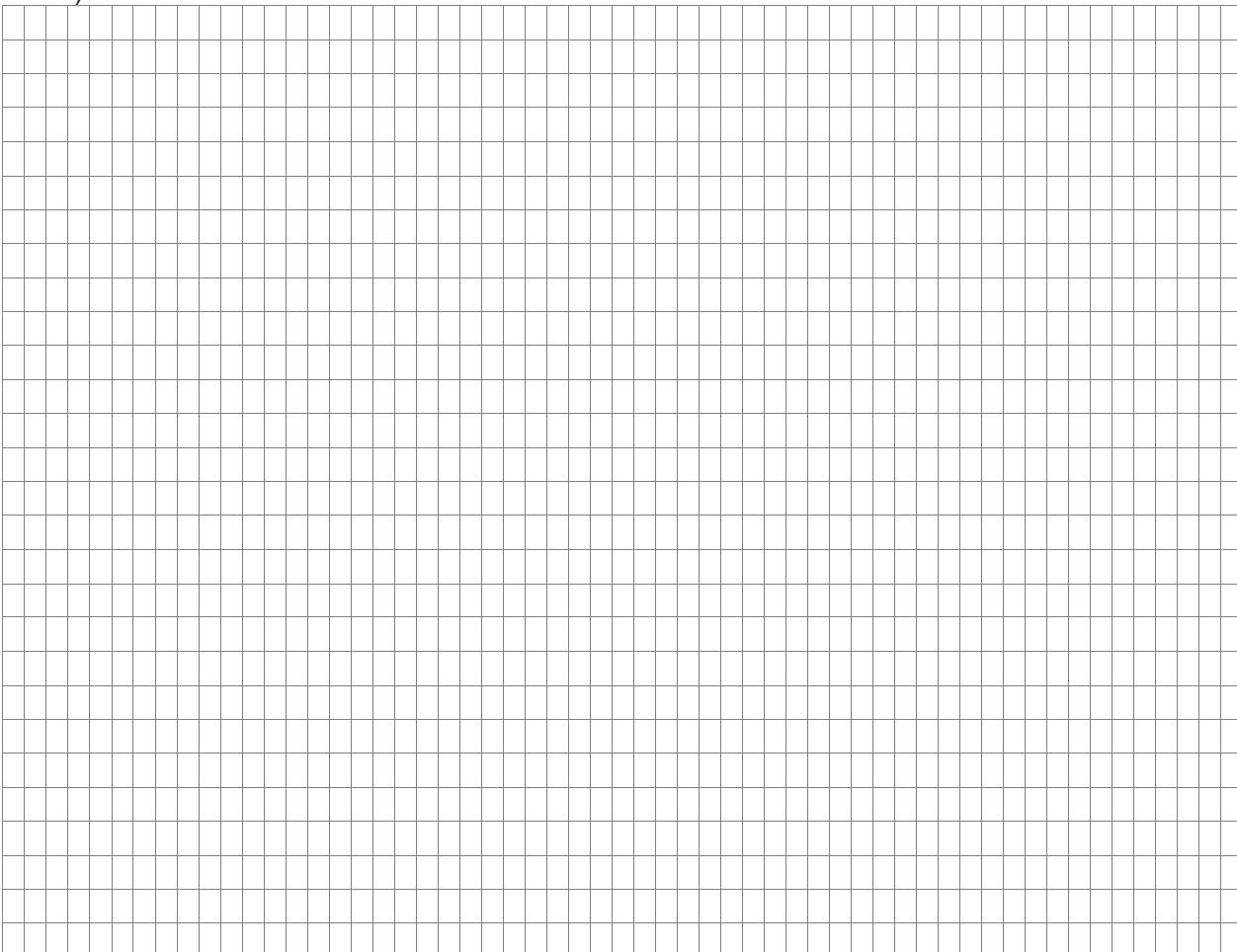
Jede so konstruierte Zahlenfolge mündet in den Zyklus 4, 2, 1 egal, mit welcher natürlichen Zahl $n > 0$ man beginnt.

Trotz zahlreicher Anstrengungen gehört diese Vermutung noch immer zu den ungelösten Problemen der Mathematik. Mehrfach wurden Preise für eine Lösung angeboten.

Aufgaben

Beschreiben Sie den Collatz-Algorithmus in den folgenden Darstellungsformen:

- a) Flussdiagramm
- b) Struktogramm
- c) Pseudocode



Aufgabe 3:

Schreiben Sie einen schrittweisen Algorithmus in Prosa (erzählender und eindeutig formulierter, aufzählender Text) für eine automatische Autowaschanlage (Waschstrasse, kein manuelles Waschen). Gehen Sie detailliert vor. Das folgende Bild soll Ihnen Ideen geben.

