Mathematik IT-1

Diskrete Mathematik für Informatik-Studiengänge

30. Oktober 2021

Prof. G. Averkov

Institut für Mathematik, Fakultät 1

Fachgebiet Algorithmische Mathematik

Brandenburgische Technische Universität Cottbus-Senftenberg

Inhaltsverzeichnis

Mitwirkende

Teile des Kapitels zu mathematischen Grundlagen enstanden aus Teilen der Tex-Mitschrit von Jonas Schulze zu meiner Vorlesung “Lineare Algebra”, die ich an der Otto-von-Guericke-Universität Mageburg hielt. Das Kapitel zu algorithmischen und Programmiergrundlagen basiert auf einem entsprechenden Kapitel zum Skript “Algorithmische Diskrete Mathematik”, welches in Zusammenarbeit mit Matthias Schymura entstanden ist.

Einleitung

1 Naturwissenschaftliche Prinzipien

1.1.

- Nur begründete bzw. verifizierte Annahmen.
- Überprüfung von Behauptungen durch Beobachtungen/Experimente und Argumente.
  - Schwerpunkt bei Mathe: Argumente
  - Experimente bei Mathe meistens billig: Beispiele, Skizzen
- Die Autorität zählt nicht ("Aber mein Lehrer hat gesagt, dass das so richtig ist"), es zählen validierte Behauptungen.
- Begriffe/Bezeichnungen werden möglichst eindeutig festgelegt (bei Mathe - extrem eindeutig), so dass man möglichst keinen Spielraum für eine Interpretation haben soll. ("War das die wahre Liebe?" - kann man die Liebe eindeutig definieren? Wenn nicht, dann kann man so eine Art Frage endlos diskutieren, ohne Ergebnis.). "Die Erde ist rund" - was heißt genau "die Erde"? Was heißt genau "rund"? Begriffsklärung sehr wichtig.
- Naturwissenschaften (Reale Welt und Modelle dazu) und Strukturwissenschaften (Modelle).
- Grundlagenwissenschaften: wenn man oft genug "Warum?" fragt, kommt man zu grundlegenden Fragen, deren Antworten einen bleibenden Wert und – auf Dauer – breite Anwendungsmöglichkeiten haben. Wichtige Frage: Warum? Warum gibt uns die pq-Formel das richtige Ergebnis? Die Antwort ist für die Mathe-Gemeinschaft interessanter als die Formel selbst. Das Interessante in der Mathematik ist genau das, was man in den Formeltafeln nicht findet.
- Modelle vs. Situationen in der realen Welt:

- Verschiedene Modelle für die selbe Situation möglich (diskrete Zeit, kontinuirliche Zeit usw.) $\rightsquigarrow$ Modelle beschreiben die reale Situation nur annähernd. Wenn wir ein Modell verstanden haben, heißt es noch nicht, dass wir die reale Situation dazu komplett verstanden haben. Das ergibt eine natürliche Unterteilung (Expert:innen mit Schwerpunkt bei Modellen, Expert:innen mit Schwerpunkt bei der realen Welt).
- Gleiche Modelle für verschiedene Situationen möglich (Wachstum von Bakterien, Wachstum in der Wirtschaft usw.) $\rightsquigarrow$ Wir können gleiche Modelle in sehr unterschiedlichen Kontexten einsetzen. Daher ist die Studie der Modelle an sich (nicht gebunden an die konkrete reale Situation) oft sinnvoll. In Mathematik/Informatik macht man genau das.

2 Mathematische Prinzipien

2.1.

- Für eine mathematische Theorie legt man Grundbegriffe und Grundbezeichnungen (eindeutig) fest.
- Auf der Basis der bereits vorhandenen Begriffen und Bezeichnungen führt man immer neue Begriffe und Bezeichnungen (eindeutig) ein.
- Mit Hilfe von vorhandenen Begriffen und Bezeichnungen werden Aussagen formuliert, die dann durch Argumentation bestätigt (= bewiesen) oder widerlegt werden.
- Wahre mathematische Beweise sind widerspruchsfrei und vollständig.
- In der Theorie interessiert man sich vor allem für noch offene Aussagen, die aktuell weder bestätigt noch widerlegt worden sind.
- Mathematik ist ungleich Rechnen. Rechnen, ohne dass man versteht, was die Rechenschritte bedeuten, ist keine mathematische Tätigkeit. Wichtig ist der Sinn hinter den Rechenschritten. Erkennt man beim Rechnen den Sinn nicht, dann ist man ein menschlicher Computer. Dabei soll man bedenken, dass echte Computer weniger Fehler und etwas schneller beim Rechnen sind :)
- Mathematik ist keine Ansammlung von Formeltafeln oder Aufzählung von mathematischen Aussagen. Formeltafeln sind Nebenprodukt davon, was man verstanden hat. Mathematische Aussagen sind ebenfalls ein Produkt, was in Mathematik viel mehr zählt sind die Begründungen (= Beweise).

Kapitel I

Mathematische Grundlagen

1 Aussagen und logische Verknüpfungen

1.1 Def. Eine Aussage ist ein Satz (eine Folge von Zeichen mit mathematischer Bedeutung), die einen eindeutigen Wahrheitswert (entweder falsch oder wahr) hat. Den Wahrheitswert kodiert man oft mit Zahlen 0 (falsch) und 1 (wahr).

1.2 Bsp.

- $2 < 1$ (falsch)
- $2 = 1$ (falsch)
- $2 > 1$ (wahr)
- 2 ist eine Primzahl (wahre Aussage, Primzahl definiert).
- 2 ist eine schöne Zahl (keine Aussage, es sei denn, die Eingeschaft einer Zahl schön zu sein, wurde definiert).
- Es gibt unendlich viele Primzahlen n, für welche $n + 2$ ebenfalls eine Primzahl ist. (eine Aussage, Wahrheitswert ist noch nicht geklärt).
- Die Gleichung $x^2 + 1 = 0$ hat keine Lösungen. (an sich keine Aussage, es sei denn, ein Kontext war vorher gegeben, in dem die Rolle von x geklärt wuirde.)
- Die Gleichung $x^2 + 1 = 0$ hat keine reellwertigen Lösungen (wahre Aussage).

1.3 Def. Seien A und B Aussagen. Dann definiert man anhand von A und B die folgenden Aussagen:

- $A \wedge B$ Konjunktion („und“) ist genau dann wahr, wenn A und B beide wahr sind.
- $A \vee B$ Disjunktion („oder“) ist genau dann falsch, wenn A und B beide falsch sind.
- $A \Rightarrow B$ Implikation ist genau dann falsch, wenn A wahr und B falsch ist.
- $A \Leftrightarrow B$ Äquivalenz ist genau dann wahr, wenn die Wahrheitswerte von A und B gleich sind.
- $A \dot{\vee} B$ ausschließende Disjunktion ist genau dann wahr, wenn die Wahrheitswerte von A und B verschieden sind.
- $\neg A$ (wir auch als $\bar{A}$ bezeichnet), Negation (Verneinung) ist genau dann wahr, wenn A falsch ist.

1.4 Bsp.

- Seien $x, y \in \mathbb{R}$. Dann gilt die Implikation: $x = y \Rightarrow x^2 = y^2$ (wahr)
- Seien $x, y \in \mathbb{R}$. Dann gilt die Implikation $x, y \in \mathbb{R}, x^2 = y^2 \Rightarrow x = y$ (falsch für $x = 1$ und $y = -1$)

1.5. Alternativbezeichnungen für $\Rightarrow$ und $\Leftrightarrow$ sind $\rightarrow$ bzw. $\leftrightarrow$.

1.6. Wenn man in Mathe-Argumenten eine Folge von Implikationen benutzt, so schreibt man auch oft kurz so etwas wie $A \Rightarrow B \Rightarrow C$. Damit meint man $(A \Rightarrow B) \wedge (B \Rightarrow C)$, d.h., aus A folgt B und aus B folgt C. Das Gleiche auch für $\Leftrightarrow$.

1.7. Die Aussagenlogik ist die Studie der logischen Vernknüpfungen von Aussagen. Dabei spielt die Natur in den Formeln verwendeten Aussagen, die man mit Symbolen bezeichnet, etwa $a, b, c, d, \ldots$, an sich keine Rolle. Alles, was zählt, ist der Wahrheitswert. Daher kann man auch $a, b, c, d \ldots$ als Variablen aus $\{0, 1\}$ auffassen, ohne dass sich an der Studie was ändert. Mehr über die Aussagenlogik erfahren wir später in diesem Kurs.

2 Mengen und Mengenoperationen

2.1 Def (Intuitive "Definition" einer Menge)**.** Eine Menge X ist durch die Eindeutige Angabe definiert, welche Objekte Elemente der Menge sind. Man schreibt in diesem Fall $x \in X$ dafür, dass das Objekt x Element der Menge X ist, und $x \notin X$ dafür, dass x kein Element der Menge X.

Mit anderen Worten: für die Angabe einer Menge X soll für jedes Objekt x geklärt sein, ob für dieses Objekt $x \in X$ oder $x \notin X$ gilt.

2.2 (Zur korrekten Definition einer Menge)**.** Unsere Definition der Menge ist etwas intuitiv und somit streng genommen keine echte Definition); sie reicht aber vorerst für unsere Zwecke völlig aus. Die genaue Definition einer Menge ist durch das Axiomensystem von Zermelo-Fraenkel gegeben. Dieses System legt Folgendes fest:

- die Existenz der leeren Mengen,
- die Bedingung für die Gleichheit von zwei Mengen
- die Möglichkeit Mengenfamilien zu vereinigen,
- die Existenz einer sogenannten Potenzmenge für eine beliebige Menge
- Fundierungsaxiom (ist etwas technisch)
- die Möglichkeit Mengen, durch eine Bedingung zu definieren.
- Ersetzungsaxiom (ist etwas technisch)

Zu den obigen Axiomen nimmt man noch zusätzlich das sogenannte Auswahlaxiom dazu. Das Axiomensystem, das auf diese Weise entsteht, wird als ZFC (Zermelo-Fraankel axioms plus Axom of Choice) abgekürzt.

2.3 Bsp (Definition einer Menge durch endliche/unendliche Auflistung)**.** Eine Weise, Mengen zu definieren, ist durch die Auflistung ihrer Elemente. Dabei stehen die geschweiften Klammern für Mengen, die drei Punkte bedeuten „usw".

- $\{1, 2, 5, 7\}$ - Menge aus den vier Elementen $1, 2, 5$ und 7.
- $\{1\}$ - Menge aus einem einzigen Element 1.
- $\{1, \{2, 5\}, \{6\}\}$ - Menge aus drei Elementen, von denen zwei Elementen selber Mengen sind.
- $\{1, 2, 3, \ldots\}$ - Menge der aller positiven ganzen Zahlen.

2.4 Def. Seien A und B Mengen. Dann heißt A eine **Teilmenge** von B, wenn jedes Element von A auch Element von B ist; wir verwenden in diesem Fall die Bezeichnung $A \subseteq B$ und nennen die Relation $\subseteq$ **Inklusion**.

Wenn A Teilmenge von B aber B keine Teilmengen von A ist, so sagt man, dass A eine **echte Teilmenge** von B ist; wir verwenden in diesem Fall die Bezeichnung $A \subsetneq B$ und nennen die Relation $\subsetneq$ eine **echte** bzw. **strikte Inklusion**.

2.5. In einigen mathematischen Quellen bezeichnet man die Inklusion als $\subset$ und nicht als $\subseteq$. Es ist schwer zu sagen, welche Bezeichnung in der Mehrheit der Quellen benutzt wird. Es gibt aber auch Quellen, in denen $\subset$ die strikte Inklusion bezeichnet. Durch die Nutzung $\subseteq$ vermeiden wir potenzielle Ambiguitäten.

2.6 Def. Mengen A und B heißen **gleich**, wenn $A \subseteq B$ und $B \subseteq A$ gilt.

2.7 (Definition einer Menge durch eine Bedingung)**.** Eine sehr verbreite Weise, Mengen zu definieren, ist durch Bedingungen, nach dem Format

$$\{\text{AUSDRUCK} : \text{BEDINGUNG}\}.$$

Der Doppelpunkt bedeutet so viel wie „sodass“oder „mit der Bedingung“. In manchen Quellen wird ein Strich an der Stelle des Doppelpunktes benutzt.

2.8 Aufg.

- Bestimmen Sie, welche der Zahlen $1, \dots, 100$ Elemente der Menge

  $$\{k^2 : k \in \mathbb{N}, k \text{ ungerade}\}$$

  sind.

- Wie viele Elemente hat die Menge $\{x \in \mathbb{R} : x^2 - 5x + 6\}$? Welche Elemente sind es genau?

2.9 Def. Die **leere Menge** ist die Menge, die keine Elemente enthält; sie wird als $\emptyset$ bzw. $\varnothing$ bezeichnet.

2.10 Def (Potenzmenge)**.** Sei X eine Menge. Dann ist die **Potenzmenge** von X die Menge aller Teilmengen von X; für diese Menge benutzen wir die Bezeichnung 2^X. Nach unserer Definition ist 2^X als

$$2^X := \{A : A \subseteq X\}.$$

gegeben.

2.11. Hier und im folgenden verwenden wir die Gleichung := mit Doppelpunkt, wenn es um neue Bezeichnungen geht, die festgelegt werden. Format:

NEUE BEZEICHNUNG := BEDEUTUNG DER BEZEICHNUNG

Die Nutzung vom Doppelpunkt in diesem Fall ist kein Muss; man kann auch durch den Begleittext verdeutlichen, dass man eine neue Bezeichnung einführt.

2.12 Aufg. Wenn X genau $n \in \mathbb{N}$ Elemente hat, wie viele Elemente hat 2^X? Was wäre Ihre Begründung dazu?

2.13. Eine weitere Bezeichnung fúr die Potenzmengen, die man in der Literatur benutzt, ist $\mathcal{P}(X)$. Ich persönlich finde 2^X einleuchtender (zumindest im Kontext der Kombinatorik, die im folgenden Kapitel diskutiert wird).

2.14. Zahlenbereiche, die Sie evtl. aus der Schule schon kennen:

$\mathbb{N} := \{1, 2, 3, \ldots\}$ die Menge der natürlichen Zahlen. Uns fehlt dort die Null, daher...

$\mathbb{N}_0 := \{0, 1, 2, \ldots\}$. Hier können wir nicht beliebig subtrahieren, daher...

$\mathbb{Z} := \{0, 1, -1, 2, -2, \ldots\}$ die Menge der ganzen Zahlen. Hier können wir nicht beliebig dividieren, daher...

$\mathbb{Q} := \{\frac{p}{q} : p \in \mathbb{Z}, q \in \mathbb{N}\}$ die Menge der rationalen Zahlen. In dieser Menge gibt es “Löcher” , die man merkt, wenn man Geometrie oder Analysis macht. Gemeint ist das Folgende: betrachtet man unendliche viele rationale Zahlen $a_1 \leq a_2 \leq a_3 \leq \cdots$ und $\cdots \leq b_3 \leq b_2 \leq b_1$ mit $a_n \leq b_n$ für jedes $n \in \mathbb{N}$, so gibt es nicht immer eine rationale Zahl x die $a_n \leq x \leq b_n$ für alle $n \in \mathbb{N}$ erfüllt. Informell: zwischen zwei Schranken a_n und b_n, die sich mit jeder “Iteration” $n \in \mathbb{N}$ immer verbessert, wird nicht immer eine rationale Zahl eingefangen. In diesem Fall kann man von einem “Loch” spricht. Durch reelle Zahlen werden solche Löcher gestopft.

$\mathbb{R}$ die Menge der reellen Zahlen. Format einer reellen Zahl: Vorzeichen $\pm$ endlich viele Stellen vor dem Komma, unendlich viele Nachkommastellen. Wir benutzen gerne die Computer-Formatierung mit Punkt an der Stelle von Komma (weil man in Mathe die Kommas gerne für viele andere Zwecke benutzt). Beispiele:

$0.00000\cdots$ ist die 0,

$1.00000\cdots$ ist die 1

$-0.99999\cdots$ ist das selbe wie $-1.000\cdots$ und ist die -1.

$0.1\underbrace{0}_{1}1\underbrace{00}_{2}1\underbrace{000}_{3}1\underbrace{0000}_{4}1\cdots$ ist eine reelle aber keine rationale Zahl. (Warum?)

$\mathbb{C}$ die Menge der komplexen Zahlen (werden wir noch einführen)

2.15. Es gelten die strikten Inklusionen. $\mathbb{N} \subsetneq \mathbb{N}_0 \subsetneq \mathbb{Z} \subsetneq \mathbb{Q} \subsetneq \mathbb{R} \subsetneq \mathbb{C}$.

2.16. Manche Quellen definieren die Menge der natürlichen Zahlen als $\{0, 1, 2, \ldots\}$, es ist mittlerweile sogar die ISO-Norm 80000-2. Aktuell spielen aber die ISO-Normen in den mathematischen Texten keine so große Rolle. Vgl. auch den ISO-Standard 31-11, in dem man z.B. $\subset$ für die strikte Inklusion reserviert.

2.17. Für Zahlenbereiche $B \in \{\mathbb{Z}, \mathbb{Q}, \mathbb{R}\}$ und ein $a \in \mathbb{R}$ benutzen wir die folgenden Bezeichnungen:

$$B_{\geq a} := \{x \in B : x \geq a\}, \qquad B_{>a} := \{x \in B : x > a\},$$
$$B_{\leq a} := \{x \in B : x \leq a\}, \qquad B_{<a} := \{x \in B : x < a\}.$$

Beispiel: $\mathbb{Z}_{\geq 2}$.

2.18 (Intervalle)**.** Seien $a, b \in \mathbb{R}$ mit $a \leq b$. Dann können Intervalle wie folgt definiert werden:

$$\begin{aligned}
[a, b] &:= \{x \in \mathbb{R} : a \leq x \leq b\} \\
(a, b) &:= \{x \in \mathbb{R} : a < x < b\} \\
(a, b] &:= \{x \in \mathbb{R} : a < x \leq b\} \\
[a, b) &:= \{x \in \mathbb{R} : a \leq x < b\} \\
[a, \infty) &:= \{x \in \mathbb{R} : x \geq a\} \\
(a, \infty) &:= \{x \in \mathbb{R} : x > a\} \\
(-\infty, a] &:= \{x \in \mathbb{R} : x \leq a\} \\
(-\infty, a) &:= \{x \in \mathbb{R} : x < a\}
\end{aligned}$$

2.19. Die Erweiterung der Zahlenbereiche zu immer größeren Bereichen ist durch den Wunsch nach einer (gewissen) Vollständigkeit motiviert.

2.20 Def. Seien A, B Mengen. Dann heißt:

- $A \cap B := \{x : (x \in A) \wedge (x \in B)\}$ **Durchschnitt** von A und B,
- $A \cup B := \{x : (x \in A) \vee (x \in B)\}$ **Vereinigung** von A und B,
- $A \setminus B := \{x : (x \in A) \wedge (x \notin B)\}$ **Mengendifferenz** von A und B,
- $A \triangle B := (A \setminus B) \cup (B \setminus A)$, **Symmetrische Differenz** von A und B.

2.21. Für eine Grundmenge X wird die Menge 2^X aller Teilmengen von X zu einer sogenannten **booleschen Algebra** der Teilmengen von X, indem man 2^X mit den Verknüpfungen $A \cap B$, $A \cap B$ und der unären Verknüpfung $\overline{A} := X \setminus A$ ausstattet.

2.22 Def. Seinen A, B Mengen. A und B heißen genau dann **disjunkt**, wenn $A \cap B = \emptyset$ gilt. In diesem Fall wird die Vereinigung von A und B eine **disjunkte Vereinigung** genannt und als $A \uplus B$ bezeichnet.

3 Abbildungen

3.1 Def. Seien X, Y Mengen. Eine **Abbildung** f von X nach Y ist eine Vorschrift, die jedem $x \in X$ genau ein Element aus Y zuordnet. Dieses Element aus Y, das dem x zugeordnet wird, wird durch $f(x)$ bezeichnet. Wenn f eine Abbildung von X nach Y ist, dann bezeichnet man das als $f : X \to Y$. Die Menge X heißt der **Definitionsbereich** von f, Y heißt der **Wertebereich** von f. Wenn der Wertebereich Y von f Teilmengen von $\mathbb{R}$ ist, so nennen wir $f : X \to Y$ eine **Funktion**.

3.2. In manchen Quellen benutzt man den Begriff Funktion als ein Synonym zum Begriff Abbildung.

3.3 Bsp.

- $f : \mathbb{R} \to \mathbb{R}$,
  $f(x) := x^2 - 2x + 7$
- $f : \mathbb{R} \setminus \{1\} \to \mathbb{R}$,
  $f(x) := \frac{1}{x-1}$

- Die Vorzeichen-Funktion sign : $\mathbb{R} \to \mathbb{R}$ wird durch die Fallunterscheidung als

$$\operatorname{sign}(x) := \begin{cases} -1 & \text{für } x < 0, \\ 0 & \text{für } x = 0, \\ 1 & \text{für } x > 0 \end{cases}$$

  definiert.

- Definitions- bzw. Wertebereiche von Abbildungen müssen keine Teilmengen der Zahlenbereiche sein.

  $f : 2^{\mathbb{N}} \to \mathbb{N}, f(A) := \min(A)$. Die Eingabe dieser Funktion ist keine Zahl sondern eine Menge, man hat z.B. $f(\{2, 7, 43\}) = 2$.

- $f : \mathbb{N} \to 2^{\mathbb{N}}, f(k) := \{1, \ldots, k\}$. Die Rückgabe dieser Abbildung ist keine Zahl sondern eine Menge, z.B. $f(3) = \{1, 2, 3\}$.

3.4. Zwei Abbildungen $f, g : X \to Y$ heißen gleich, falls $f(x) = g(x)$ für alle $x \in X$ gilt.

3.5. Für Mengen X und Y, bezeichnet man als Y^X die Menge aller Abbildungen von X nach Y.

3.6 Aufg. Aus wie vielen Abbildungen besteht die Menge $\{1, 2, 3\}^{\{1,2\}}$? Zählen Sie diese Abbildungen auf? Was ist mit $\{1, 2\}^{\{-1,0,1\}}$?

3.7 Def. Seien X, Y, A, B Mengen mit $A \subseteq X$ und $B \subseteq Y$. Sei $f : X \to Y$. Dann heißt $f(A) := \{f(x) : x \in A\}$ das Bild von A bzgl. f und $f^{-1}(B) := \{x \in X : f(x) \in B\}$ das Urbild von B bzgl. f.

3.8 Bsp. Sei $f : \mathbb{R} \to \mathbb{R}$ mit $f(x) := x^2$ für alle $x \in \mathbb{R}$.

- $f([1, 2]) = [1, 4]$ ↗ Abb. 1
- $f^{-1}([1, 4]) = [1, 2] \cup [-2, -1]$ ↗ Abb. 2
- $$\begin{aligned} f^{-1}([-7, 8]) &= \{x \in \mathbb{R} : f(x) \in [-7, 8]\} \\ &= \{x \in \mathbb{R} : -7 \leq f(x) \leq 8\} \\ &= \{x \in \mathbb{R} : -7 \leq x^2 \leq 8\} \\ &= \{x \in \mathbb{R} : x^2 \leq 8\} \\ &= \{x \in \mathbb{R} : |x| \leq \sqrt{8}\} \\ &= [-\sqrt{8}, \sqrt{8}] \end{aligned}$$

3.9 Def. Seien X, Y Mengen und sei $f : X \to Y$. Dann heißt f:

- injektiv, falls für alle $x_1, x_2 \in X : x_1 \neq x_2$ die Bedingung $f(x_1) \neq f(x_2)$ gilt.
- surjektiv, falls für jedes $y \in Y$ ein $x \in X$ mit der Eigenschaft $f(x) = y$ existiert.
- bijektiv, falls f injektiv und surjektiv ist.

3.10 Bsp. Untersuche folgende Funktionen auf Bijektivität:

- $f : \mathbb{R} \to \mathbb{R}, f(x) := x^2$ für alle $x \in \mathbb{R}$
  surjektiv ? nein, $-1 \neq f(x)$ für alle $x \in \mathbb{R}$
  injektiv ? nein, $f(x) = f(-x)$ für alle $x \in \mathbb{R}$
- $f : \mathbb{R} \to [0, +\infty), f(x) := x^2$ für alle $x \in \mathbb{R}$
  surjektiv ? ja
  injektiv ? nein (analog)
- $f : \mathbb{R} \setminus \{0\} \to \mathbb{R}, f(x) = \frac{1}{x}$ für alle $x \in \mathbb{R}$
  surjektiv ? nein, 0 wird nicht angenommen
  injektiv ? ja
- $f : \mathbb{R} \to \mathbb{R}, f(x) = 2x + 3$ für alle $x \in \mathbb{R}$
  bijektiv ? ja

3.11 Def. Seien X, Y Mengen und sei $f : X \to Y$ bijektiv. Die Abbildung, die jedem $y \in Y$ das eindeutige $x \in X$ mit $f(x) = y$ zuordnet, heißt die Umkehrabbildung von f und wird durch f^{-1} bezeichnet.

3.12 Aufg. Was ist die Umkehrfunktion von $f : \mathbb{R} \to \mathbb{R} : f(x) := 2x + 3$?

3.13 Def. Seien X, Y, Z Mengen, $f : X \to Y$ und $g : Y \to Z$. Dann heißt $g \circ f : X \to Z$ mit $(g \circ f)(x) := g(f(x))$ für alle $x \in X$ die Komposition von g und f.

3.14 Bsp. Seien $f : \mathbb{R} \to \mathbb{R} : f(x) = 2x + 3$ für alle $x \in \mathbb{R}$ und $g : \mathbb{R} \to \mathbb{R} : g(x) = x^2$ für alle $x \in \mathbb{R}$. Dann ist $(f \circ g)(x) = 2x^2 + 3$ und $(g \circ f)(x) = (2x + 3)^2$.

3.15 Def. Sei X eine Menge. Dann heißt die Abbildung $\mathrm{id}_X : X \to X$ mit $\mathrm{id}_X(x) := x$ für alle $x \in X$ die identische Abbildung auf X. Man schreibt auch häufig id, wenn X nicht angegeben werden muss.

3.16. Seien X, Y Mengen und sei $f : X \to Y$ bijektiv. Dann gilt

- $f \circ f^{-1} = \mathrm{id}_Y$,
- $f^{-1} \circ f = \mathrm{id}_X$.

4 Vereinigung und Durchschnitt einer indexierten Mengenfamilie

4.1 Def. Seien I, X Mengen und sei $A : I \to 2^X$. Man schreibt auch in diesem Fall A_i statt $A(i)$ für $i \in I$, $(A_i)_{i\in I}$ ist eine Familie (Schar) von Teilmengen von X.

Für die Familie $(A_i)_{i\in I}$ definiert man

$$\text{den Durchschnitt} \qquad \bigcap_{i\in I} A_i := \{x \in X : x \in A_i \text{ für alle } i \in I\},$$

und

$$\text{die Vereinigung} \qquad \bigcup_{i\in I} A_i := \{x \in X : x \in A_i \text{ für ein } i \in I\}.$$

4.2 Bsp. Sei $\alpha \in (0, \pi)$ und $v_0 > 0$ (↗ Abb. 3). K_α ist die Flugbahn beim Auswurf eines Objekts mit der Anfangsgeschwindigkeit v_0 unter dem Winkel α zu Erde.

$$K_\alpha := \{(x, y) \in R^2 : x = \cos(\alpha)t, y = \sin(\alpha)t - \frac{gt^2}{2}, y \geq 0, t \geq 0\}$$

$$(K_\alpha)_{\alpha\in(0,\pi)}$$

$$\bigcap_{\alpha\in(0,\pi)} K_\alpha = \{(0, 0)\}$$

$$\bigcup_{\alpha\in(0,\pi)} K_\alpha = \text{alle Werte unter der Parabel (↗ Abb. 4)}$$

5 Summen und Produkte

5.1 Def. Eine Menge X heißt endlich, falls $X = \emptyset$ oder falls eine bijektive Abbildung von $\{1, \ldots, n\}$ nach X existiert mit $n \in \mathbb{N}$. Der Wert n heißt die Anzahl der Elemente (Kardinalität) von X und wird durch $|X|$ bezeichnet. Man setzt die Kardinalität von $\emptyset$ gleich 0. Bei einer unendlichen Mengen X setzt man $|X| = \infty$.

5.2. $|X|$ ist wohldefiniert, d.h. eine Menge kann nicht zwei unterschiedliche Kardinalitäten haben.

5.3 Def. Sei X eine nichtleere endliche Menge. Dann kann X als $X = \{x_1, \ldots, x_n\}$ dargestellt werden mit $x_i \neq x_j$ für alle $i, j \in \{1, \ldots, n\}$ mit $i \neq j$.

Für eine Abbildung $f : X \to \mathbb{R}$ definiert man

$$\sum_{x \in X} f(x) := f(x_1) + \ldots + f(x_n),$$
$$\prod_{x \in X} f(x) := f(x_1) \cdot \ldots \cdot f(x_n).$$

Im Fall $X = \emptyset$ definiert man für $f : X \to \mathbb{R}$ und $\sum\limits_{x \in X} f(x) = 0$ und $\prod\limits_{x \in X} f(x) = 1$.

In der Summe $\sum\limits_{x \in X} f(x)$ heißt $f(x)$ der *Summand* und im Produkt $\prod\limits_{x \in X} f(x)$ heißt $f(x)$ der *Faktor*.

5.4. Die Summe und das Produkt über eine Menge X sind wohldefiniert: die beiden Werte sind von der Nummerierung $x_1, \ldots, x_n$ der Elemente von X unabhängig. Das liegt daran, dass $+$ und $\cdot$ beide kommutative Operationen sind.

5.5. $\sum\limits_{i=a}^{b}$ benutzt man als eine kurze Schreibweise für $\sum\limits_{i \in \{a, \ldots, b\}}$. Im entarteten Fall $a > b$ ist $\sum\limits_{i=a}^{b}$ eine Summe über die leere Menge.

5.6 Bsp. Für jedes $n \in \mathbb{N}$ gilt $S := \sum_{i=1}^{n} i = \frac{1}{2} n(n+1)$. Die Abbildung $i \mapsto n+1-i$ ist eine Bijektion von $\{1, \ldots, n\}$ nach $\{1, \ldots, n\}$. Daher gilt $S = \sum_{i=1}^{n} (n+1-i)$. Daraus ergibt sich

$$2S = S + S = \sum_{i=1}^{n} i + \sum_{i=1}^{n} (n+1-i) = \sum_{i=1}^{n} (i+n+1-i) = \sum_{i=1}^{n} (n+1) = n(n+1).$$

Das ergibt die gewünschte Darstellung $S = \frac{1}{2} n(n+1)$.

6 Tupel und Kreuzprodukte

6.1 Def (Paare und das Kreuzprodukt)**.** Für beliebige Objekte a, b kann man das *(geordnete) Paar* (a, b) definieren. Das Paar (a, b) besteht aus der ersten Komponente a und der zweiten Komponente b. Die Gleichheit $(a, b) = (c, d)$ von Paaren wird durch die Gleichheit $a = c$ und $b = d$ der jeweiligen Komponenten definiert. Für Mengen A, B definiert man das *Kreuzprodukt* (wird auch das *kartesische Produkt* genannt) $A \times B$ als die Menge

$$A \times B := \{(a, b) \,:\, a \in A,\ b \in B\}$$

aller Paare bei denen die erste Komponente in A und die zweite Komponente in B ist.

6.2 Def (Tupel und Kreuzprodukt)**.** Komplett analog zu (geordneten) Paaren definiert man auch (geordnete) Tripel (a, b, c), (geordnete) Quadrupel (a, b, c) und noch allgemeiner (geordnete) n-Tupel $(x_1, \ldots, x_n)$ mit $n \in \mathbb{N}_0$. Dem entsprechend betrachtet man auch das Kreuzprodukt $A \times B \times C$ von drei Mengen, das Kreuzprodukt $A \times B \times C \times D$ von vier Mengen und allgemein auch das Kreuzprodukt

$$X_1 \times \ldots \times X_n := \{(x_1, \ldots, x_n) \,:\, x_1 \in X_1, \ldots, x_n \in X_n\}$$

von Mengen $X_1, \ldots, X_n$.

Das Element x_i mit $i \in \{1, \ldots, n\}$ im n-Tupel $(x_1, \ldots, x_n)$ heißt die i-te *Komponente* des Tupels.

Für eine Menge X führt man die Bezeichnung

$$X^n := \underbrace{X \times \ldots \times X}_{n \text{ mal}} = \{(x_1, \ldots, x_n) : x_1, \ldots, x_n \in X\}$$

ein.

6.3 Aufg. Was stellen die folgenden Kreuzprodukte geometrisch dar? Zeichnen Sie diese Mengen:

- $[0, 1] \times [0, 2]$, $\{0\} \times [0, 2]$ und $\{0, 1\} \times \{0, 2\}$
- $[0, 1]^3$, $[0, 1]^2 \times \{0\}$, $[0, 1]^2 \times \{1\}$ und $\{0\}^2 \times [0, 1]$.

Hinweis: Geometrische Darstellung von Teilmengen von $\mathbb{R}^n$ für $n \in \{2, 3\}$ im kartesischen Koordinatensystem wird in den Kursen IT-2 und IT-3 nützlich sein.

6.4 Aufg. Beim bekannten Spiel "Schiffe versenken" lässt sich das Spielfeld als die Menge $\{1,\dots,10\}^2$ beschreiben, weil jedes Kästschen eines Feldes durch die Angabe von einem Paar (x,y) mit $x,y \in \{1,\dots,10\}$ beshcrieben werden können. Eine Platzierung eines 4er-Schiffs lässt sich dann als eine vierelementige Teilmengen von $\{1,\dots,10\}^2$ beschreiben. Z.B.:

$$\{(1,2),(1,3),(1,4),(1,5)\}$$

und

$$\{(5,6),(6,6),(7,6),(8,6)\}$$

sind zwei mögliche Platzierungen. Beschreiben Sie die Menge aller möglichen Platzierungen eines 4er-Schiffs mit Hilfe der mathematischen Bezeichnungen, die Sie oben gelernt haben.

7 Prädikate und Quantoren

7.1 Def. Sei X Menge. Dann heißt $P : X \to \{\text{falsch}, \text{wahr}\}$ *Prädikat* auf X.

7.2. Informell beschrieben ist ein Prädikat eine Aussage über ein variables Element x aus X. Dabei hängt im Allgemeinen der Wahrheitswert der Aussage von der Wahl von x ab. Sonst lässt sich ein Prädikat auf X als eine Eigenschaft eines variablen Elements $x \in X$ beschreiben, die entweder vorhanden oder nicht vorhanden ist. Zum Beispiel: $P(n) :=$"n ist gerade" als Eigenschaft einer natürlichen Zahl $n \in \mathbb{N}$. Diese Eigenschaft ist bei $n = 123$ nicht vorhanden und bei $n = 2020$ vorhanden.

7.3 Bsp. $P : \mathbb{N} \to \{\text{falsch}, \text{wahr}\}$ mit $P(k) :=$ „$k(k+1)$ ist durch 3 teilbar".

7.4 Bsp. $P : \mathbb{R}^2 \to \{\text{falsch}, \text{wahr}\}$ mit $P(x,y) :=$ „$x^2 + y^2 \leq 1$" ist die Eigenschaft "der Punkt (x,y) hat den Abstand höchstens 1 zum Punkt $(0,0)$".

7.5. Für eine gegebene Menge X hat man eine natürliche Bijektion zwischen der Menge $\{\text{falsch}, \text{wahr}\}^X$ aller Prädikate auf X und der Menge 2^X aller Teilmengen von X. Jedes Prädikat $P : X \to \{\text{falsch}, \text{wahr}\}$ erzeugt die Menge $\{x \in X : P(x)\}$ und jede Menge $A \subseteq X$ erzeugt das Prädikat $P(x) :=$ „$x \in X$".

7.6 Def. $\forall\, x \in X : P(x)$ für ein Prädikat P auf eine Menge X steht für die Aussage „die Bedingung $P(x)$ gilt für alle $x \in X$." Das Symbol $\forall$ heißt das *Allgemeinheitsquantor* (Bedeutung: für $\forall$lle).

$\exists\, x \in X : P(x)$ bezeichnet die Aussage „die Bedingung $P(x)$ gilt für ein $x \in X$.“
$\exists$ heißt *Existenzquantor* (Bedeutung: es $\exists$xistiert).

7.7. Negierung von quantifizierten Aussagen erfolgt auf die folgende naheliegende Weise:

- $\overline{\forall\, x \in X : P(x)} \Leftrightarrow \exists\, x \in X : \overline{P(x)}$
- $\overline{\exists\, x \in X : P(x)} \Leftrightarrow \forall\, x \in X : \overline{P(x)}$

7.8. $\forall$ und $\exists$ lassen sich kombinieren. Seien X, Y Mengen. Wenn man ein Prädikat P auf $X \times Y$ hat, so kann man dafür die Aussagen wie

$$\forall x \in X\, \exists\, y \in Y : P(x, y),$$

und

$$\exists x \in X\, \forall y \in Y : P(x, y)$$

usw. einführen.

7.9. In der vorigen Bemerkung ist die Reihenfolge des Quantifizierens relevant. Sei X eine Menge von Personen und A eine Menge von Adressen im Stadtteil Sandow. Die Aussage

$$\forall x \in X\, \exists a \in A\, :\, x \text{ wohnt unter der Adresse } a.$$

lautet, dass alle Personen aus X irgendwo in Sandow wohnen. Die Aussage

$$\exists a \in A\, \forall x \in X\, :\, x \text{ wohnt unter der Adresse } a.$$

lautet dagegen, dass alle Personen aus X unter einer und der selben Adresse in Sandow wohnen (z.B. als Wohngemeinschaft). Man sieht, die letztere Aussage ist eine stärkere Bedingung.

7.10 Bsp. Hier ein Beispiel einer Definition aus der Analysis, die man kompakt mit Quantoren und Prädikaten einführen kann.

Sei $(a_n)_{n\in\mathbb{N}}$ Folge reeller Zahlen (mit anderen Worten: $a : \mathbb{N} \to \mathbb{R}$) und sei $\alpha \in \mathbb{R}$. Dann heißt α Grenzwert von $(a_n)_{n\in\mathbb{N}}$, falls das Folgende gilt:

$$\forall\, \epsilon \in \mathbb{R}_{>0}\, \exists\, N \in \mathbb{R}\, \forall\, n \in \mathbb{N} : ((n \geq N) \Rightarrow (|a_n - \alpha| < \epsilon))$$

8 Relationen

8.1 Def. Seien X, Y Mengen. Dann heißt eine Teilmenge R von $X \times Y$ eine (binäre) *Relation* zwischen X und Y. Bei $X = Y$, heßt R eine (binäre) Relation auf X.

8.2. Da man Prädikate auf $X \times Y$ mit Teilmengen von $X \times Y$ identifizieren kann, lassen sich Relationen auch als Prädikate auf $X \times Y$ auffassen.

8.3. Wenn für $x \in X$ und $y \in Y$ die Bedingung $(x, y) \in R$ gilt, so schreibt man $x\,R\,y$. Das bedeutet: x steht in der Relation R zu y.

8.4 Bsp.

- $X = \{f_1, f_2, f_3, f_4\}$ - Menge von Fahrzeugen
  $Y = \{\text{Ersatzrad}, \text{Radio}, \text{Navi}, \text{Automatik}\}$ - Menge von Features von Fahrzeugen. Hier

	Ersatzrad	Radio	Navi	Automatik
f_1	ja	ja	ja	ja
f_2	ja	ja	ja	nein
f_3	nein	nein	ja	ja
f_4	nein	ja	ja	nein

  Diese Tabelle legt eine Relation auf $X \times Y$ fest.

- $\leq, <, \geq, >$ sind binäre Relationen auf $\mathbb{R}$
- Sei X Menge. Dann ist $\subseteq$ ist eine binäre Relation auf 2^X.
- Für $a, b \in \mathbb{N}$ schreibt man $a|b$, wenn b durch a ohne Rest teilbar ist. Dies ist eine binäre Relation auf $\mathbb{N}$.

8.5 Def. Sei X Menge und $\sim$ eine Relation auf X. Dann heißt $\sim$ eine **Äquivalenzrelation**, falls:

1. $\sim$ ist *reflexiv*, d.h. $x \sim x$ für alle $x \in X$.
2. $\sim$ ist *symmetrisch*, d.h. $x \sim y$ ist äquivalent zu $y \sim x$ für alle $x \in X$.
3. $\sim$ ist *transitiv*, d.h. aus $x \sim y$ und $y \sim z$ folgt $x \sim z$ für alle $x, y, z \in X$.

Für eine Äquivalenzrelation $\sim$ auf einer Menge X und ein $x \in X$ heißt

$$[x]_\sim := \{y \in X : x \sim y\}$$

die Äquivalenzklasse von x bzgl. $\sim$. Die Menge aller Äquivalenzklassen von $\sim$ ist

$$X/\sim := \{[x]_\sim : x \in X\}.$$

8.6 Bsp.

- Sei V endliche Menge und sei $\binom{V}{2} := \{\{u, v\} : u, v \in V, u \neq v\}$. Das Paar (V, E) mit $E \subseteq \binom{V}{2}$ heißt *Graph* mit Kantenmenge V und Knotenmenge E.

  $G = (V, E), G = \{1, \ldots, 6\}, E = \{\{1, 2\}, \{2, 3\}, \{3, 4\}, \{4, 1\}, \{1, 3\}, \{5, 6\}\}$

  Für $a, b \in V$ heißt b von a aus *erreichbar* (im Graphen $G = (V, E)$), falls ein $k \in \mathbb{N}_0$ und Elemente $u_0, \ldots, u_k \in V$ existieren mit $u_0 = a$, $u_k = b$ und $\{u_i, u_{i+1}\} \in E$ für alle $i \in \mathbb{N}_0$ mit $i < k$.

  Die Erreichbarkeit ist eine Äquivalenzklasse auf V. Die Äquivalenzklassen (Zusammenhangskomponenten) für dieses Beispiel sind $\{1, 2, 3, 4\}$ und $\{5, 6\}$.

- Sei $m \in \mathbb{N}$. Für $a, b \in \mathbb{Z}$ sagt man, dass a kongruent zu b modulo m ist, falls $a - b \in m\mathbb{Z}$, wobei $m\mathbb{Z} := \{mz : z \in Z\}$.

  Schreibweise: $a \equiv b \mod m$.

  Die Kongruenz modulo m ist eine Äquivalenzrelation auf $\mathbb{Z}$.

- Sei $\sim$ Relation auf $\mathbb{Z} \times \mathbb{N}$, definiert durch $(a, b) \sim (c, d)$ für $a, c \in \mathbb{Z}, b, d \in \mathbb{N}$, wenn $ad = bc$ gilt.

  Diese Relation ist eine Äquivalenzrelation (Aufgabe).

  D.h. jede rationale Zahl ist eine Äquivalenzklasse von diesem $\sim$.

8.7 Def. Eine Menge X mit einer binären Relation $\succeq$ darauf heißt Poset (partiell geordnete Menge), wenn für alle $x, y, z \in X$ folgendes gilt:

- $x \preceq x$ (Reflexivität)
- $x \preceq y,\ y \preceq z \Rightarrow x \preceq z$ (Transitivität).
- $x \preceq y,\ y \preceq x \Rightarrow x = y$ (Antisymmetrie).

Die binäre Relation $\succeq$ heißt in diesem Fall die partielle Ordnung auf X.

8.8 Def. Wenn für ein Poset $(X, \succeq)$ für alle $x, y \in X$, die Bedingung $x \succeq y$ oder

die Bedingung $y \succeq x$ erfüllt ist, so nennt man $(X, \succeq)$ eine total geordnete Menge und $\succeq$ eine totale Ordnung auf X.

8.9 Bsp. Beispiele von Posets.

- 2^X mit Inklusion.
- $\mathbb{N}$ mit Teilbarkeit.
- Substring-Relation auf Strings. Ist A eine endliche nichtleere Menge so heißt

$$A^* = \bigcup_{n \in \mathbb{N}_0} A^n$$

die Menge der Strings über dem Alphabet A. Die Menge $\{0,1\}^*$ heißt die Menge der binären Strings. In diesem Kontext schreibt man oft $a_1 \cdots a_n$ an der Stelle von $(a_1, \ldots, a_n)$, etwa $10011 \in \{0,1\}^*$ und nicht $(1,0,1,1) \in \{0,1\}^*$.

8.10 Def. Für $n \in \mathbb{N}$ ist eine n-stellige Relation auf Mengen $X_1, \ldots, X_n$ eine Teilmenge $R \subseteq X_1 \times \cdots \times X_n$.

8.11 Bsp. Betrachten wir eine Tabelle, in welcher die Besucher:innen eines Hotels durch die Angaben **Name, Zimmer, Checkin-Datum, Checkout-Datum** geführt werden. Ist S die Menge aller Strings und D die Menge aller Daten, so kann mann die Tabelle als eine 4-stellige Relation $R \subseteq S \times S \times D \times D$ auffassen. Die Bedingung $(p, z, d_1, d_2) \in R$, dass (p, z, d_1, d_2) sich in der Relation R befinden, bedeutet, dass die Person p am Tag d_1 im Zimmer z untergebracht wurde und am Tag d_2 das Hotel verlassen hat.

Wie man an diesem Beispiel sieht, sind die Tabellen eine Möglichkeit Relationen R durch eine Aufzählung (durch die Zeilen einer Tabelle) zu beschreiben.

9 Beweisansätze

9.1 Direkter Beweis

9.1. Eine grobe Beschreibung eines direkten Beweises ist wie folgt. Ein direkter Beweis einer Implikation $a \Rightarrow b$ ist ein Beweis der auf Implikationen basiert, die von a ausgehen und zum b führen.

9.2. Man kann für $x \in \mathbb{R}$ die Äquivalenz:

$$x^2 - 5x + 6 = 0 \qquad \Longleftrightarrow \qquad x \in \{2, 3\}$$

als zwei Implikationen ausschreiben und anschließend folgendermaßen direkt verifizieren. Ist $x^2 - 5x + 6 = (x-2)(x-3)$. Also folgt aus $(x-2)(x-3) = 0$, dass $x - 2 = 0$ oder $x - 3 = 0$ gilt. Im ersten Fall erhält man aus $x - 2 = 0$, dass $x = 2$ ist. Im zweiten Fall erhält man aus $x - 3$, dass $x = 3$ ist. Folglich hat man $x \in \{2, 3\}$. Umgekehrt: ist $x \in \{2, 3\}$, so hat man im Fall $x = 2$ die Gleichheiten $x^2 - 5x + 6 = 2^2 - 5 \cdot 2 + 6 = 4 - 10 + 6 = 0$ und im Fall $x = 3$ die Gleichheiten $x^2 - 5x + 6 = 3^2 - 5 \cdot 3 + 6 = 9 - 15 + 6 = 0$. Aus $x \in \{2, 3\}$ folgt also $x^2 - 5x + 6$.

9.2 Indirekte Beweise

9.3. Ein Widerspruchsbeweis ist ein Beweis, bei dem man eine Aussage a, die man zeigen möchte, durch die Herleitung der Implikation $\overline{a} \Rightarrow$ falsch bestätigt bzw. eine Implikation $a \Rightarrow b$, die man zeigen möchte, durch die Herleitung der Implikation $a \wedge \overline{b} \Rightarrow$ falsch bestätigt.

In den beiden Fällen erhält man aus einer Annahme einen sogenannten Widerspruch, d.h., eine falsche Aussage. Den Widerspruch erhält man oft in der Form $c \wedge \overline{c}$ für eine Aussage c.

9.4 Lem. Für $t \in \mathbb{N}$ seien $p_1, \ldots, p_t \in \mathbb{N}$ Zahlen mit $p_i \geq 2$ für alle $i \in \{1, \ldots, t\}$ und sei $n := p_1 \cdots p_t + 1$. Dann ist n durch keine der Zahlen $p_1, \ldots, p_t$ teilbar.

Beweis. Angenommen, n wäre durch ein p_i mit $i = 1, \ldots, t$ teilbar. Da aber das Produkt $p_1 \cdots p_t$ durch p_i teilbar ist, ist $1 = n - p_1 \cdots p_t$ ebenfalls durch p_i teilbar. Wir haben also gezeigt, dass 1 durch die ganze Zahl p_i, mit $p_i \geq 2$, teilbar ist. Das ist ein Widerspruch, der uns die Behauptung unseres Lemmas bestätigt. □

9.5. Eine Beweis durch Kontraposition ist der Beweis der Implikation $a \Rightarrow b$ dadurch, dass man die Implikation $\overline{b} \Rightarrow \overline{a}$ bestätigt. Ein Beweis durch Kontraposition und der Widerspruchsbeweis sind miteinander verwandt, denn einen Beweis durch Kontraposition kann man in einen Widerspruchsbeweis konvertieren.

9.6. Beweise durch Kontraposition und Widerspruch nennt man *indirekt.*

9.7 Lem. Sei $a \in \mathbb{N}$. Dann sind die folgenden Aussagen äquivalent:

(a) a ist gerade.

(b) a^3 ist gerade.

Beweis. Wir zeigen (a) $\Rightarrow$ (b) direkt. Ist a gerade, so hat a die Form $a = 2k$ mit $k \in \mathbb{N}$. Somit ist $a^3 = (2k)^3 = 8k^3$ ebenfalls gerade.

Die Implikation (b) $\Rightarrow$ (a) können wir durch die Kontraposition herleiten: wir zeigen also $\neg$ (a) $\Rightarrow \neg$ (b). Wenn a ungerade ist, so hat a die Form $a = 2k + 1$ mit $k \in \mathbb{N}_0$. Somit ist $a^3 = (2k+1)^3 = (2k)^3 + 3(2k)^2 + 3(2k) + 1 = 2(4k^3 + 6k^2 + 6k) + 1$ eine ungerade Zahl. □

9.8 Thm. *Die Zahl $\sqrt[3]{2}$ ist nicht rational.*

Beweis. Angenommen, $\sqrt[3]{2}$ wäre rational. Dann hätte die Zahl die Form $\sqrt[3]{2} = \frac{a}{b}$ mit $a, b \in \mathbb{N}$. Darüber hinaus können wir annehmen, dass a und b nicht beide gerade sind, denn sonst kann man a und b, solange sie beide gerade sind, durch 2 teilen, wodurch sich a und b um Faktor zwei verkleinern. Es ist klar, dass dieser Prozess nach endlich vielen Schritten terminiert.

$\sqrt[3]{2} = \frac{a}{b}$ folgt $2b^3 = a^3$. Es folgt also, dass a^3 gerade ist. Dann ist aber nach Lemma 9.7 die Zahl a gerade ist und somit die Form $a = 2k$ mit $k \in \mathbb{N}$ hat. Dann ist $2b^3 = a^3 = (2k)^3 = 8k^3$, woraus $b^3 = 4k^3$ folgt. Die Zahl b^3 ist also gerade. Nach Lemma 9.7, die wir nun zur Zahl b anwenden können, ist die Zahl b ebenfalls gerade. Wir haben also gezeigt, dass a und b beide gerade sind. Unsere Annahme war aber, dass a oder b ungerade ist. Dieser Widerspruch zeigt, dass die Zahl $\sqrt[3]{2}$ nicht rational ist. □

9.3 Vollständige Induktion

9.9 Thm (Vollständige Induktion, Version 1)**.** *Sei P ein Prädikat auf $\mathbb{N}$. Dann sind die folgenden Bedingungen äquivalent:*

(a) $P(n)$ gilt für alle $n \in \mathbb{N}$.

(b) $P(1)$ gilt und, aus $P(n)$ folgt $P(n+1)$, für alle $n \in \mathbb{N}$.

Beweis. Die Implikation (a) $\Rightarrow$ (b) ist klar: $P(1)$ ist erfüllt und da $P(n)$ und $P(n+1)$ beide Wahr ist die Implikation $P(n) \Rightarrow P(n+1)$ für jedes n eine wahre Aussage.

Nun zeigen wir (a) $\Rightarrow$ (b) durch Kontraposition. Angenommen, (a) ist nicht erfüllt. Dann gibt ein $n \in \mathbb{N}$ für welches $P(n)$ falsch ist. Wir fixieren das kleinste solche $n \in \mathbb{N}$. Ist unser $n = 1$ so, ist (b) nicht erfüllt, weil $P(1)$ nicht erfüllt ist. Ist $n > 1$ so ist (b) nicht erfüllt, weil $P(n)$ falsch und $P(n-1)$ wahr ist, wodurch die Implikation $P(n-1) \Rightarrow P(n)$ nicht erfüllt ist. □

9.10. Beim Verwenden von Theorem 9.9 unterteilt sich die Argumentation in die folgende Schritte.

- Induktionsanfang (IA): man verifiziert, dass $P(1)$ gilt.
- Induktionsvoraussetzung (IV): man macht die Annahme: sei $n \in \mathbb{N}$ und sei die Aussage $P(n)$ erfüllt.
- Induktionsschritt (IS): man folgert $P(n+1)$ aus der Induktionsvoraussetzung.

9.11 Thm. *Für jedes $n \in \mathbb{N}$ gilt*

$$\sum_{i=1}^{n} i = \frac{1}{2} n(n+1).$$

Beweis. Das Prädikat mit dem wir uns in dieser Aussage befassen ist die Gleichung

$$\sum_{i=1}^{n} i = \frac{1}{2} n(n+1)$$

die von einem variablen $n \in \mathbb{N}$ abhängig ist.

Diese Formel ist für $n = 1$ erfüllt, denn $\sum_{i=1}^{1} i = 1$ und $\frac{1}{2} 1 \cdot (1+1) = 1$.

Sei nun $n \in \mathbb{N}$ ein beliebiger Wert, für welche die Formel $\sum_{i=1}^{n} i = \frac{1}{2} n(n+1)$ erfüllt ist. Wir zeigen, dass die Formel mit $n+1$ an der Stelle von n ebenfalls erfüllt ist. Es gilt

$$\sum_{i=1}^{n+1} = \sum_{i=1}^{n} i + (n+1),$$

da wir in der Summe den Summanden zum Index $i = n+1$ abspalten können. Nach der Induktionsvoraussetzung ist $\sum_{i=1}^{n} i = \frac{1}{2} n(n+1)$. Somit hat man

$$\sum_{i=1}^{n+1} = \frac{1}{2} n(n+1) + (n+1) = \frac{1}{2} (n+1)(n+2).$$

Zusammenfassend: Unsere Formel gilt für $n = 1$ und wenn unsere Formel für ein $n \in \mathbb{N}$ erfüllt ist, so ist sie auch mit $n+1$ an der Stelle von n erfüllt. Aus Theorem 9.9 folgt, dass unsere Formel für jedes $n \in \mathbb{N}$ erfüllt ist. □

9.12 Bsp. Sei $q \in \mathbb{R} \setminus \{1\}$ und $n \in \mathbb{N}_0$. Dann gilt $\sum_{i=1}^{n} q^i = \frac{q^{n+1}-1}{q-1}$. Zeigen Sie diese Formel durch die Induktion über n. Hier ist der Induktionsanfang $n = 0$.

9.13 Bsp. Finden Sie eine Formel für $\sum_{i=1}^{n} i q^i$ mit $q \in \mathbb{R} \setminus \{1\}$ und $n \in \mathbb{N}_0$ und beweisen Sie diese Formel durch Induktion.

9.14. Durch Induktion lassen sich nicht nur Gleichungen herleiten. Es gibt viele verschiedene Situationen in der Mathematik (und insbesondere diskreter Mathematik), in denen man durch die Induktion Aussagen verifizieren kann.

9.15 Thm. *$n \leq 2^n$ gilt für alle $n \in \mathbb{N}$.*

Beweis. Diese Ungleichung kann man mit der Verwendung Ihrer Schulkenntnisse aus der Analysis herleiten. Der folgende Beweis durch die Induktion ist aber elementarer.

Die Ungleichung gilt für $n = 1$, denn $1 \leq 2^1$. Sei nun $n \in \mathbb{N}$ ein Wert, für welchen $n \leq 2^n$ gilt. Im Induktionsschritt sollen wir nun $n+1 \leq 2^{n+1}$ herleiten. Da wir $n \leq 2^n$ voraussetzen, gilt $n + 1 \leq 2^n + 1$, daher reicht es zu verifizieren, dass $2^n + 1 \leq 2^{n+1}$ erfüllt ist. Das letztere ist Äquivalent zur Ungleichung $2^n \geq 1$, die trivialerweise für $n \in \mathbb{N}$ erfüllt ist. □

9.16 Aufg. Zeigen Sie, dass $100n \leq 2^n$ für alle $n \in \mathbb{N}$ mit $n \geq 10$ erfüllt ist.

9.17 Thm (Vollständige Induktion, Version 2)**.** *Sei P ein Prädikat auf $\mathbb{N}$. Dann sind die folgenden Bedingungen äquivalent:*

(a) $P(n)$ gilt für alle $n \in \mathbb{N}$.

(b) es gilt $P(1)$ und, für jedes $n \in \mathbb{N}$, gilt die Implikation

$$P(1) \wedge \cdots \wedge P(n) \Rightarrow P(n+1).$$

Beweis. Es gibt zwei einfache Weisen, diese Version der vollständigen Induktion herzuleiten. Zum einen kann man den Beweis von Theorem 9.9 sehr geringfügig modifizieren, um dieses Theorem herzuleiten. Zum anderen kann man die Behauptung von Theorem 9.9 für das Prädikat $Q(n) := P(1) \wedge \cdots \wedge P(n)$ benutzen. □

9.18 Thm (Primfaktorzerlegung - Existenz)**.** *Für jedes $n \in \mathbb{N}$ existieren Primzahlen $p_1, \ldots, p_t$ ($t \in \mathbb{N}_0$), deren Produkt gleich n ist, d.h.:*

$$n = \prod_{i=1}^{t} p_i.$$

9.19. Kommentar zur vorigen Behauptung: man hat $\prod_{i=1}^{0} p_i = 1$ im Fall $t = 0$ und $\prod_{i=1}^{1} p_i = p_1$ im Fall $t = 0$.

Beweis. Die Behauptung "es existieren $t \in \mathbb{N}_0$ Primzahlen $p_1, \ldots, p_t$ mit $n = \prod_{i=1}^{t} p_i$" ist wahr für $n \in \{1, 2\}$. Denn $n = 1$ ist Produkt $1 = \prod_{i=1}^{0} p_i = 1$ und im Fall $n = 2$ ist $2 = \prod_{i=1}^{1} p_i$ mit $p_1 := 1$.

Sei nun $n \in \mathbb{N}$ mit $n \geq 3$ so, dass jede Zahl $a \in \{1, \ldots, n-1\}$ Produkt von endlich vielen Primzahlen ist (im Sinne der Behauptung). Ist n Primzahl, so gilt die Behauptung mit $t = 1$ und $p_1 = n$. Ist n keine Primzahl, so besitzt n einen Teiler $a \in \{2, \ldots, n-1\}$. Es folgt $n = ab$ mit $b = n/a \in \mathbb{N}$ und $b \leq \frac{n}{2} \leq n - 1$. Die Anwendung der Induktionsvoraussetzung zu a und b ergibt, dass man a sowie b als Produkt von Primzahlen darstellen hat. Es gilt also

$$a = \prod_{i=1}^{r} u_i,$$
$$b = \prod_{i=1}^{s} v_i.$$

mit $r, s \in \mathbb{N}$ für geweisse Primzahlen $u_1, \ldots, u_r, v_1, \ldots, v_s$ (hier ist weder r noch s gleich 0, denn $a, b \geq 2$). Dann ist n Produkt von $t = r + s$ Primzahlen $p_1, \ldots, p_t$ mit $p_i = u_i$ für $i \in \{1, \ldots, r\}$ und $p_i = v_{i-r}$ für $i \in \{r+1, \ldots, r+s\}$. □

9.20 (Fallunterscheidung)**.** Ein weiteres verbreitetes Element eines Beweises ist die Fallunterscheidung. Im vorigen Beweis haben wir z.B. zwischen den Fällen, dass n eine Primzahl und n keine Primzahl ist, unterschieden. In jedem der beiden Fällen gaben wir ein eigenes Argument, wieso n in die Primfaktoren zerlegbar ist.

9.21. Im vorigen Beweis setzen wir das Muster $P(1) \wedge \cdots \wedge P(n) \Rightarrow P(n+1)$ aus Theorem 9.17 als $P(1) \wedge \cdots \wedge P(n-1) \Rightarrow P(n)$ um. Das heißt, in Theorem 9.17 werden die schon abgearbeiteten Werte als $1, \ldots, n$ bezeichnet und der nächste Wert als $n + 1$. In unserer Umsetzung werden die abgearbeiteten Werte als $1, \ldots, n-1$ bezeichnet und der nächste Wert als n.

9.22 (Vollständige Induktion und Algorithmen)**.** Induktionsbeweise haben oft eine algorithmische Interpretation und können zu iterativen oder rekursiven Algorithmen konvertiert werden. In diesem Fall, kann man sich die folgende Umsetzung des vorigen Beweises als Algorithmus vorstellen. Nehmen wir an, wir wollen für alle Zahlen $1, \ldots, N$ für ein gegebenes $N \in \mathbb{N}$, $N \geq 2$, deren Zerlegung in Primfaktoren berechnen. Dann können wir folgendermaßen vorgehen. Wir führen für jedes $n \in \{1, \ldots, N\}$, eine Liste $L[i]$ der Primfaktoren von $1, \ldots, N$ ein. Für $n = 1$ ist dies eine leere Liste. Angenommen, die Liste sei bereits für die Zahlen $1, \ldots, n-1$ erzeugt. Dann können wir die Liste $L[n]$ anhand der bereits vorhanden Listen $L[1], \ldots, L[n-1]$ so generieren. Wir testen, ob n Primzahl ist. Ist das der Fall so erzeugen wir $L[n]$ als Liste aus

einer einzigen Zahl n. Ist n keine Primzahl, so Faktorisieren wir n als $n = ab$ mit $a, b \in \{2, \ldots, n-1\}$ und erzeugen dann $L[n]$ durch das Zusammenfügen der Listen $L[a]$ und $L[b]$.

```
def smallest_factor_which_is_at_least_two(n):
    """
        der kleinster Faktor einer natuerrlichen Zahl n>=2, der mind. 2
            ist.

        Ist die Rueckgabe n, so ist n Primzahl.
    """
    assert(n>=2)
    for i in range(2,n):
        if n % i ==0 : # teilbar durch i?
            return i
    # HINWEIS: diese Funktion laesst sich verschnellern
    # (im Fall einer Eingabe, die eine Primzahl ist)
    # AUFGABE: Ueberlegen Sie sich, wie man die Funktion verschnellern
        kann
    return n

def prime_factorizations(N):
    """
        Primfaktorzerlegungen aller Zahlen von 1 bis N, fuer N>=1
    """
    assert(N>=1)
    Z={1:[]} # 1 ist Produkt von 0 Primzahlen
    for n in range(2,N+1):
        a=smallest_factor_which_is_at_least_two(n)
        if a==n:
            # n Primzahl
            Z[n]=[n] # n ist Produkt einer Primzahl (die Zahl n selbst)
        else:
            Z[n]=sorted(Z[a] + Z[n/a]) # die Primaftorzerlegung von n
                setzt sich
            # aus Primfaktorzerlegungen von a und n/a zusammen
    return Z

for n,f in prime_factorizations(25).items():
    print("Alle Faktoren von {}: {}".format(n,f))
```

9.4 Ein Beispiel und ein Kommentar zur Fallunterscheidung

9.23 Bsp (Ein Beispiel zur Fallunterscheidung)**.** Sei M die Menge der Nutzer:innen eines Sozialnetzwerks. Auf M ist eine binäre Relation "befreundet sein" definiert, welche symmetrisch ist. Wir zeigen, dass in jeder 6-elementigen Teilmenge P von M drei Nutzer:innen existieren, die entweder gegenseitig befreundet sind oder gegenseitig nicht befreundet sind.

Wir fixieren zuerst ein beliebiges $p \in P$. Bezüglich p zerlegt sich die 5-elementige Menge $P \setminus \{p\}$ in die Menge A der jenigen $a \in P \setminus \{p\}$, die mit p befreundet sind, und der Mengen B der jenigen $b \in P \setminus \{p\}$, die mit p nicht befreundet sind. Da $A \cup B = P \setminus \{p\}$ insgesamt 5 Elemente hat, hat eine der beiden Mengen A oder B mindestens 3 Elemente: denn hätten A oder B beide weniger als 3 Elemente, so hätte $A \cup B$ insgesamt höchstens 4 Elemente, was der Tatsache widerspricht, dass $A \cup B = P \setminus \{p\}$ 5 Elemente hat.

> *Fall 1:* A hat mindestens 3 Elemente, d.h., p ist mit mindestens drei Personen aus $P \setminus \{p\}$ befreundet.
>
> > *Fall 1a:* in A findet man zwei befreundete Personen a', a''. Dann ist $\{p, a', a''\}$ eine 3-elementige Teilmenge von P, mit den Personen, die befreundet sind.
> >
> > *Fall 1b:* in A sind keine zwei Personen befreundet. Dann sind alle Personen aus A nicht gegenseitig befreundet. In A hat man also drei Personen die nicht gegenseitig befreundet sind.
>
> *Fall 2:* B hat mindestens 3 Elemente, d.h., p ist mit mindestens drei Personen aus $P \setminus \{p\}$ nicht befreundet.
>
> > *Fall 2a:* in B findet man zwei Personen b', b'', die nicht befreundet sind. Dann ist $\{p, b', b''\}$ eine 3-elementige Teilmengen von P, mit den Personen, die nicht befreundet sind.
> >
> > *Fall 2b:* in B sind alle Personen gegenseitig befreundet. Dann hat man in B drei Personen, die gegenseitig befreundet sind.

In jedem der möglichen Fälle, haben wir die Existenz von drei Personen in P nachgewiesen, die entweder gegenseitig befreundet sind oder gegenseitig nicht befreundet sind.

9.24 (Fallunterscheidung mit Computer)**.** In manchen Beweisen in Mathematik benötigt man so viel Fallunterscheidung (etwa zur Abarbeitung einer endlichen Anzahl von Sonderfälle), dass man es nicht mehr schafft, all die Fälle manuell zu behandeln. Dann holt man sich oft Computer zur Hilfe.

10 Schnupperstunde in Algebra

10.1 Was ist Algebra?

10.1. Algebra ist die Theorie algebraischer Strukturen. Während man in der Schule mit einer relativ kleiner Anzahl algebraischer Strukturen wie $(\mathbb{R}, +, \cdot)$ oder dem Vektorraum $\mathbb{R}^3$ arbeitet, befasst man sich in Algebra mit verschiedenen Kategorien algebraischer Strukturen, wie z.B. Halbgruppen, Gruppen, Ringe, Körper und Vektorräume.

Man entwickelt auch Mittel, neue/eigene algebraische Strukturen anzulegen. Wenn man diesen Prozess mit der Programmierung vergleicht, so ist der Prozess sehr ähnlich zur Entwicklung eigener Datenstrukturen (im Gegensatz zur Nutzung der standardmäßig vorhandenen Datenstrukturen).

10.2 (Algebraische Struktur)**.** Eine algebraische Struktur ist in der Regel eine Menge A, die mit einer oder mehreren Verknüpfungen ausgestattet ist. In den allermeisten Fällen sind die Verknüpfungen, die man betrachtet, binär: sie sind Abbildungen $* : A \times A \to A$. Für solche Abbildungen schreibt man dann $a * b$ an der Stelle von $*(a, b)$. Sehr oft handelt es sich auch um die Verknüpfungen, für welche (zumindest) das Assoziativgesetz $a * (b * c) = (a * b) * c$ erfüllt ist.

10.3 (Polymorphismus in Algebra)**.** Man benutzt oft zum Bezeichnen der Verknüpfungen (bzw. der Verknüpfung) einer algebraischen Struktur die Symbole + (Plus) und $\cdot$ (Mal). Hierbei meint man dann die Plus-Operation bzw. die Mal-Operation innerhalb der gegebenen algebraischen Struktur A. Das heißt, diese Operationen müssen + und/oder $\cdot$ innerhalb einer algebraische Struktur A nicht unbedingt mit Operation + und $\cdot$ innerhalb der Menge $\mathbb{R}$ der reellen Zahlen etwas zu tun haben. Das bedeutet: genau so, wie Symbole $a, b, c, d, \ldots$ in Mathematik kontextabhängig sind (können verschiedene Bedeutung in verschiedenen Kontexten haben), sind auch die Bezeichnungen wie + und $\cdot$ kontextabhängig (bzw. Strukturabhängig) und können so, wie man es sich wünscht, eingeführt werden. Wenn man also + in der Struktur A hat, so ist das streng genommen $+_A$ – die Plusoperation aus der Struktur A – man schreibt aber einfach nur + und nimmt stillschweigend an, dass es aus dem Kontext klar ist, welche Struktur A gemeint ist. Die Nutzung der selben Bezeichnungen für verschiedene Operationen nennt man in der Programmierung der Polymorphismus.

10.4 Bsp. Für $n \in \mathbb{N}$ heißt die Menge S_n aller bijektiven Abbildungen von $\{1, \ldots, n\}$ nach $\{1, \ldots, n\}$ mit der Multiplikation

$$(\sigma \cdot \tau)(i) := \sigma(\tau(i))$$

die symmetrische Gruppe. Was eine (allgemeine) Gruppe ist, wird in IT-2 diskutiert.

10.5 Bsp. Die algebraische Struktur $\mathbb{F}_2$, welche man als Menge $\{0,1\}$ mit den binären Operationen

$+$	0	1
0	0	1
1	1	0

und

$\cdot$	0	1
0	0	0
1	0	1

einführt, ist ein sogenannter binärer Körper. Die Bezeichnungen $+$, $\cdot$, 0 und 1, die wir hier verwenden, sind polymorph.

Wir meinen $+_{\mathbb{F}_2}, \cdot_{\mathbb{F}_2}$, $0_{\mathbb{F}_2}$ und $1_{\mathbb{F}_2}$ schreiben aber in unserem Kontext von $\mathbb{F}_2$ vereinfachend $+, \cdot, 0, 1$.

Der binäre Körper spielt in der Kodierungstheorie und der Kryptographie eine wichtige Rolle.

10.2 Kommutativer Ring

10.6 Def. Eine Menge R mit zwei binären Verknünfungen $+, -$ und zwei verschiedenen ausgezeichneten Elementen $0, 1 \in R$ heißt kommutativer Ring, wenn für alle $a, b, c \in R$ Folgendes erfüllt ist:

- $a + b = b + a$ und $a \cdot b = b \cdot a$
- $a + 0 = a$ und $a \cdot 1 = a$
- $(a + b) + c = a + (b + c)$ und $a \cdot (b \cdot c) = (a \cdot b) \cdot c$
- Zu jedem a gibt es ein eindeutiges Element aus R, das man als $-a$ bezeichnet, für welches $a + (-a) = 0$ erfüllt ist.
- $a \cdot (b + c) = a \cdot b + a \cdot c$

10.7 Aufg. Ist R kommutativer Ring mit 1, dann gilt $a \cdot 0 = 0$ für alle $a \in R$. Zeigen Sie das.

10.8 Bsp.

- $(\mathbb{N}, +, \cdot)$ kein Ring.
- $(\mathbb{N}_0, +, \cdot)$ (immer noch) kein Ring.
- $(\mathbb{Z}, +, \cdot)$ ein kommutativer Ring.

- $(\mathbb{Q}, +, \cdot)$ ein kommutativer Ring.
- $(\mathbb{R}, +, \cdot)$ ein kommutativer Ring.
- $(\mathbb{C}, +, \cdot)$ ein kommutativer Ring.

10.3 Körper

10.9 Def. Eine Menge K mit zwei binären Verknüpfungen $+$ und $\cdot$ heißt Körper, wenn K bzgl. $+$ und $\cdot$ kommutativer Ring ist und darüber hinaus für jedes $a \in K \setminus \{0\}$ ein eindeutiges Element $a^{-1} \in K$ existiert, für welches $a \cdot a^{-1} = 1$ gilt.

10.10 Bsp.

- $(\mathbb{F}_2, +, \cdot)$
- Führen Sie auf einer dreielementigen Menge $\{0, 1, a\}$ die Verknüpfungen $+$ und $\cdot$ so ein, dass die Menge mit diesen Verknüpfungen zu einem Körper wird.
- $(\mathbb{Z}, +, \cdot)$ kein Körper, da in $\mathbb{Z} \setminus \{0\}$ nichts außer -1 und 1 invertierbar ist.
- $(\mathbb{Q}, +, \cdot)$ ein Körper.
- $(\mathbb{R}, +, \cdot)$ ein Körper.
- $(\mathbb{C}, +, \cdot)$ ein Körper.

10.11 Def. Ein Körper K heißt algebraisch abgeschlossen, wenn für jede Wahl von $d \in \mathbb{N}$ und alle $a_d \in K \setminus \{0\}, a_{d-1}, \ldots, a_0 \in K$ die Gleichung

$$a_d x^d + a_{d-1} x^{d-1} + \cdots + a_0 = 0$$

mindestens eine Lösung x aus K besitzt. Eine Gleichung wie oben nennt man Polynomgleichung vom Grad d mit Koeffizienten in K.

10.12 Bsp.

- $\mathbb{Q}$ ist nicht algebraisch abgeschlossen, vgl. die Gleichung $x^2 - 2 = 0$, mit den Koeffizienten $-2, 0, 1 \in \mathbb{Q}$, die keine Lösung x in $\mathbb{Q}$ besitzt.
- $\mathbb{R}$ ist nicht algebraisch abgeschlossen, vgl. die Gleichung $x^2 + 1 = 0$ mit den Koeffizienten $1, 0, 1 \in \mathbb{R}$, die keine Lösung x in $\mathbb{R}$ besitzt.

10.13 Def. Sind A und B Mengen mit $A \subseteq B$ und $*_A : A \times A \to A$ und $*_B : B \times B \to B$ binäre Verknüpfungen, so nennt man $*_B$ Erweiterung von $*_A$ und $*_A$ Einschränkung von $*_B$ auf A, wenn $x *_A y = x *_B y$ für alle $a, b \in A$ erfüllt ist (mit anderen Worten: $*_B$ wirkt genau so wie $*_A$ innerhalb von A).

10.14 Def. Sind $(F, +, \cdot)$ und $(K, +, \cdot)$ Körper mit $F \subseteq K$, bei denen $+$ und $\cdot$ von K Erweiterungen von $+$ bzw. $\cdot$ auf F sind, so nennt man den Körper K eine Erweiterung des Körpers F.

10.15 Bsp. $\mathbb{R}$ ist Erweiterung von $\mathbb{Q}$. Es gibt aber viele Körper dazwischen. Zum Beispiel ist

$$\mathbb{Q}[\sqrt{2}] := \left\{ a + \sqrt{2}b \,:\, a, b \in \mathbb{Q} \right\}$$

ebenfalls ein Körper. Es gilt $\mathbb{Q} \subsetneq \mathbb{Q}[\sqrt{2}] \subsetneq \mathbb{R}$. Wie sieht das inverse eines Elements aus $\mathbb{Q}[\sqrt{2}] \setminus \{0\}$ aus?

10.16 Thm. *Jeder Körper besitzt eine algebraisch abgeschlossene Körpererweiterung.*

10.17. Es gilt sogar eine stärkere Aussage: jeder Körper eine (im einem bestimmten Sinn) minimale algebraisch abgeschlossene Körpererweiterung.

10.4 Der Körper der komplexen Zahlen

10.18 Def. Die Menge $\mathbb{C}$ der komplexen Zahlen führen wir als die Menge der formalen Ausdrücke der Form $x + y\,\mathbf{i}$ mit $x, y \in \mathbb{R}$ ein. Hierbei ist $\mathbf{i}$ ein formales Element, für welches wir $\mathbf{i}^2 := -1$ festlegen. Das Element $\mathbf{i}$ nennt man die *imaginäre Einheit* oder die *Wurzel aus* -1. Die Menge der reellen Zahlen $\mathbb{R}$ wird als eine Teilmenge von $\mathbb{C}$ aufgefasst, indem man $x \in \mathbb{R}$ als $x + y\,\mathbf{i}$ mit $y = 0$ schreibt.

Nach diesen Festlegungen lassen sich die Operationen $+$ und $\cdot$ vom Körper $\mathbb{R}$ der reellen Zahlen auf $\mathbb{C}$ auf eine eindeutige Weise erweitern, wenn man fordert, dass $\mathbb{C}$ mit Operationen $+$ und $\cdot$ ein kommutativer Ring sein soll, vgl. dazu die Gesetze für einen kommutativen Ring. (Wie wir in Kürze sehen werden, ist $(\mathbb{C}, +, \cdot)$ sogar ein Körper.) Die Addition und Multiplikation führen wir also auf die folgende Weise eingeführt:

$$\begin{aligned}(x_1 + y_1\mathbf{i}) + (x_2 + y_2\mathbf{i}) &:= (x_1 + y_1) + (y_1 + y_2)\mathbf{i} \\ (x_1 + y_1\mathbf{i}) \cdot (x_2 + y_2\mathbf{i}) &:= (x_1x_2 - y_1y_2) + (x_1y_2 + x_2y_1)\mathbf{i},\end{aligned}$$

für $x_1, x_2, y_1, y_2 \in \mathbb{R}$.

Ist $z = x + y\mathbf{i}$ mit $x, y \in \mathbb{R}$ so führen, wir den Realteil von z als $\mathrm{Re}(z) := x$ und den Imaginärteil von z als $\mathrm{Im}(z) := y$ ein; die Zahl $\overline{z} = x - y\mathbf{i}$ nennen wir komplex konjugiert zu z; den Wert $|z| = \sqrt{x^2 + y^2}$ nennen wir den Betrag von z.

10.19 Aufg. Berechnen Sie $(3 + 2\mathbf{i})(5 - \mathbf{i})$, indem Sie eine Darstellung dieser Zahl als $x + y\mathbf{i}$ mit $x, y \in \mathbb{R}$ bestimmen.

10.20. In Algebra werden oft Strukturen formal nach "eigenen Vorgaben" eingeführt. Bei der Definition von komplexen Zahlen sieht man ein Beispiel dafür.

10.21 Thm. $\mathbb{C}$ *ist ein algebraisch abgeschlossener Körper.*

Beweis. Dass $(\mathbb{C}, +, \cdot)$ ein kommutativer Ring ist, lässt sich direkt verifizieren (Aufgabe).

Um zu zeigen, dass $(\mathbb{C}, +, \cdot)$ sogar ein Körper ist, muss man verifizieren, dass jedes $z = x + y\mathbf{i}$ mit $x, y \in \mathbb{R}$ mit $|z| \neq 0$ ein inverses Element in $\mathbb{C}$ besitzt. Es stellt sich heraus, dass man das inverse Element z^{-1} als $z = \frac{1}{|z|^2}\overline{z}$ beschreiben kann. Mit der Verwendung der dritten binomischen Formel erhalten wir

$$zz^{-1} = \frac{z\bar{z}}{|z|^2} = \frac{(x + y\mathbf{i})(x - y\mathbf{i})}{x^2 + y^2} = \frac{x^2 - (y\mathbf{i})^2}{x^2 + y^2} = \frac{x^2 - y^2\mathbf{i}^2}{x^2 + y^2} = \frac{x^2 + y^2}{x^2 + y^2} = 1.$$

Dass der Körper $(\mathbb{C}, +, \cdot)$ algebraisch abgeschlossen ist, ist ziemlich bemerkenswert. Bedenken Sie, dass wir nur die imaginäre Einheit $\mathbf{i}$ eine formale Lösung der Polynomgleichung $z^2 + 1 = 0$ in einem unbekannten z eingeführt haben. Die Behauptung über die algebraische Abgeschlossenheit ist, dass wird durch diese Ergänzung für eine beliebige Polynomgleichung von einem positiven Grad (un d mit Koeffizienten in $\mathbb{C}$) eine Lösung in $\mathbb{C}$ finden. Um diese Behauptung herzuleiten braucht man wissen aus der Analysis (wir geben also an dieser Stelle keinen Beweis). □

10.22 Aufg. Zeigen Sie $|u \cdot v| = |u| \cdot |v|$ für alle $u, v \in \mathbb{C}$. **Hinweis:** Am besten zeigt man $|u \cdot v|^2 = |u|^2 \cdot |v|^2$, um die Wurzeln zu vermeiden.

10.5 Exkurs: Trigonometrie

10.23 Thm (Der Satz des Pythagoras)**.** *In einem rechtwinkligen Dreieck seien a und b die Längen der am rechten Winkel anliegenden Seiten und c die Länge*

der dem rechten Winkel gegenüberliegenden Seite. Dann gilt $c^2 = a^2 + b^2$.

10.24 Kor. Der Abstand zwischen dem Punkt $(0,0) \in \mathbb{R}^2$ und dem Punkt $(x,y) \in \mathbb{R}^2$ ist gleich $\sqrt{x^2+y^2}$.

Beweis. Im Fall $x = 0$ oder $y = 0$ ist die Behauptung klar, da sich der Punkt (x,y) auf einer der beiden Koordinatenachsen befindet. Im Fall $x \neq 0$ und $y \neq 0$ ist das Dreieck mit den Ecken $(0,0), (x,0), (x,y)$ rechtwinklig und die Längen der am rechten Winkel anliegenden Seiten sind $|x|$ und $|y|$. Nach dem Satz des Pythagoras ergibt das die Länge $\sqrt{|x|^2+|y|^2} = \sqrt{x^2+y^2}$ für den Abstand zwischen $(0,0)$ und (x,y). □

10.25 Def. Sei $c \in \mathbb{R}^2$ und $\rho > 0$. Dann heißt die Menge aller Punkte, den zum Punkt c Abstand $\rho > 0$ haben **Kreis** mit Zentrum in c vom Radius $\rho > 0$.

10.26. Aus Korollar 10.24 folgt: die Menge

$$\left\{(x,y) \in \mathbb{R}^2 \,:\, x^2+y^2 = 1\right\}$$

ist Kreis vom Radius 1 mit Zentrum in $(0,0)$. Diese Menge nennen wir im folgenden **den Einheitskreis**.

10.27 (Über Radianten und Grade.)**.** Im alten Babylonien dachte man, das Jahr wäre 360 Jahre lang (das stimmt nicht, wie wir jetzt wessen). Daher teilte man den Jahreskreis in 360 Teile auf, die den Tagen entsprechen. Ein Grad steht daher für einen Tag im babylonischen Jahreskreis. Das zeigt, dass die Herkunft der Messung der Winkel in Graden nicht mathematisch ist. Sie ist anthropologisch: sie hängt mit dem Planeten Erde zusammen, auf dem wir uns befinden, und mit den Babylonier:innen, die bei der Bestimmung der Anzahl der Tage im Jahr sich ein Wenig verschätzten. Dennoch hat sich die Messung mit 360 Graden für den vollen Winkeln bis jetzt erhalten. Das liegt vielleicht daran, dass einige für uns interessante Winkel mit Graden durch eine ganze Zahl darstellbar sind (90°, 60°, 30°).

Die Messung mit Radianten ist eine dimensionslose Messung und sie ist intrinsisch mathematisch. Man nimmt einen Kreis mit dem Radius 1 und misst Winkel durch die Längen der Bögen dieses Kreises. Dabei bezeichnet man die Länge einer Hälfte des Einheitskreises als π und nennt die Zahl π die Kreiszahl. Diese Zahl π ist etwas größer als 3 (das sieht man, wenn man in den Einheitskreis ein reguläres Sechseck einschreibt).

Ein Grad ist nichts Anderes als

$$1^\circ := \frac{\pi}{180} = \frac{2\pi}{360}.$$

Wenn man Winkel in Radianten misst, kann man etwa 1.2 Radianten aber auch einfach nur 1.2 sagen, denn die Einheit Radiant ist dimensionslos.

An sich gibt es an der Messung der Winkel mit Graden nichts Falsches. Dieser Kommentar dient einfach nur dazu, darauf hinzuweisen, dass die Zahlen wie 360 und 180 in Bezug auf die Winkelmessung keine mathematische sondern eine anthropologische Natur haben.

10.28 (Kosinus, Sinus und Co.)**.** Man betrachte eine kreisförmige Radrennbahn mit Zentrum im Punkt $(0, 0)$ vom Radius 1. Diese Bahn ist nach dem Satz des Pythagoras durch die Gleichung $x^2 + y^2 = 1$ beschrieben. Nun legen wir den Punkt $(1, 0)$ dieser Bahn als den Startpunkt fest. Von diesem Punkt aus kann man nun Strecken einer beliebigen Länge zurücklegen. Wie lang die Strecke ist und ob man sich im Gegenuhrzeiger oder im Uhrzeigersinn bewegt wird durch eine Zahl $\alpha \in \mathbb{R}$ notiert. Der Betrag von α gibt die Länge der Strecke an, die man zurücklegen will. Das Vorzeichen von α gibt an, ob man sich im Gegenuhrzeigersinn oder im Uhrzeigersinn bewegt: bei einem positiven Vorzeichen - im Gegenuhrzeigersinn und bei einem negativen Vorzeichen - im Uhrzeigersinn. Für jedes $\alpha \in \mathbb{R}$ erhält man einen Punkt (x, y), in dem man sich nach dem Zurücklegen der vorgegebenen Strecke in die vorgegebene Richtung landet. Die x-Komponente dieses Punkts nennt man den Kosinus von α und die y-Komponente Sinus von α. Bezeichnungen dazu sind:

$$x = \cos\alpha$$
$$y = \sin\alpha$$

Sonst betrachtet man den Tangens

$$\tan\alpha = \frac{\sin\alpha}{\cos\alpha}$$

den Kotangens

$$\cot\alpha = \frac{\cos\alpha}{\sin\alpha}$$

Ist $0 \leq \alpha \leq \pi$, so sagt man bei $x = \cos\alpha$, dass α Arcus Kosinus von x ist. Man schreibt dann $\alpha = \arccos x$. Mit anderen Worten ist arccos die Umkehrfunktion der vom Kosinus, wenn man den Kosinus als Funktion $[0, \pi] \to [-1, 1]$ auffasst. Ist $-\pi \leq \alpha \leq \pi$, so sagt man bei $y = \sin\alpha$, dass α Arcus Sinus von y ist. Man schreibt dann $\alpha = \arcsin y$. Mit anderen Worte ist arcsin die Umkehrfunktion der Einschränkung vom Sinus, wenn man den Sinus als Funktion $[-\pi, \pi] \to [-1, 1]$ auffasst.

10.29 (Kosinus und Sinus im Taschenrechner)**.** Wenn Studierende den Kosinus und Sinus (unter anderem für sehr einfache Werte α) im Taschenrechner berechnen, so sieht man, dass es immer wieder dazu kommt, dass ihre Ergebnisse falsch sind. Das liegt daran, dass man in vielen Taschenrechnern eine Umschaltung zwischen Grad und Radianten hat. Ist der Taschenrechner auf Radianten eingestellt, so berechnet er die eigentlichen Kosinus und Sinus, wie sie in Mathematik (und in den meisten Programmiersprachen) zu finden sind. Ist der Taschenrechner auf Grade eingestellt, so berechnet er die Funktionen $t \mapsto \cos(\frac{\pi}{180}t)$ und $t \mapsto \sin(\frac{\pi}{180})t$ an der Stelle von cos und sin. Übrigens: in Excel wird die Funktion $t \mapsto \frac{\pi}{180}t$, die oben in cos und sin eingesetzt wurde, das Bogenmaß von t genannt.

10.30. Die vielen Formeln, die man für den Kosinus und Sinus und andere trigonometrische Funktionen hat, lassen sich im Rahmen der linearen Algebra (IT-3) viel besser verstehen.

10.6 Eulersche Formel

10.31 Thm. *Jede komplexe Zahl $z \in \mathbb{C}$ besitzt eine Darstellung als*

$$z = \rho(\cos\phi + \mathbf{i}\sin\phi)$$

mit $\rho \in \mathbb{R}_{\geq 0}$ und $\phi \in \mathbb{R}$. Hierbei gilt $\rho = |z|$. Bei $z \neq 0$, ist ϕ eindeutig durch z bis auf das addieren eines ganzzahligen Vielfachen von 2π definiert.

Beweis. WARNUNG: Der nachfolgende Beweis und unsere Definition von cos und sin entspricht nicht ganz den mathematischen Standards, solange wir den Begriff Länge (eines Bogens) und Orientierung (einer Kurve), auf den wir uns bei der Einführung von cos und sin beziehen, nicht mathematisch formal definiert haben. Wir verlassen uns also auf Intuition und darauf, dass man (später) den Begriff Länge mathematisch korrekt einführen kann (solche Begriffe führt man in der Analysis ein). Es gibt auch einen formalen nicht-geometrischen Zugang zum Kosinus und Sinus (dieser Zugang ist aber nicht wirklich intuitiv, sodass man dadurch nicht wirklich versteht, was Kosinus und Sinus eigentlich sind).

Da jede komplexe Zahl $z = x + y\mathbf{i}$ eindeutig durch $x, y \in \mathbb{R}$ gegeben ist, kann man z als einen Punkt $(x, y) \in \mathbb{R}^2$ visualisieren. Die Visualisierung von $\mathbb{C}$ auf diese Weise nennt man die gaußsche Zahlenebene. Dabei werden 1 und $\mathbf{i}$ als die zueinander senkrechte Vektoren $(1, 0)$ und $(0, 1)$ dargestellt. Man sieht, dass die Menge $K := \{z \in \mathbb{C} : |z| = 1\} = \{x + \mathbf{i}y : x^2 + y^2 = 1\}$ als der Einheitskreis mit Zentrum in $0 \in \mathbb{C}$ und dem Radius 1 in der gaußschen Zahlenebene darstellbar ist.

Existenz: Ist $z \neq 0$,so ist $z/|z|$ ist ein Punkt im Kreis K und so hat z die Darstellung $z/|z| = \cos\phi + \mathbf{i}\sin\phi$ für ein $\phi \in \mathbb{R}$ nach unserer Beschreibung von cos und sin. Es folgt also, dass $z = \rho(\cos\phi + \mathbf{i}\sin\phi)$ mit $\rho = |z|$ gilt. Im Fall $z = 0 \in \mathbb{C}$ kann man $\rho = 0$ und ein beliebiges ϕ fixieren.

Eindeutigkeit: Ist $z = \rho(\cos\phi + \mathbf{i}\sin\phi)$ mit $\rho \in \mathbb{R}_{\geq 0}$ und $\phi \in \mathbb{R}$ so gilt $|z| = |\rho(\cos\phi + \mathbf{i}\sin\phi)| = \rho|\cos\phi + \mathbf{i}\sin\phi| = \rho\sqrt{\cos^2\phi + \sin^2\phi} = \rho$. Ist $z \neq 0$, so ist $z/|z|$ der Punkt $\cos\phi + \mathbf{i}\sin\phi$ auf Einheitskreis K. Der Punkt $\cos\phi + \mathbf{i}\sin\phi$ im Kreis K ändert sich nicht, wenn man zum Wert von ϕ ein ganzzahliges Vielfaches von 2π dazu addiert, weil der Kreis K die Länge 2π hat. So besteht die Möglichkeit als ϕ einen Wert aus $[0, 2\pi)$ zu wählen.

Da K die Länge 2π hat, ist jeder Punkt eindeutig durch die Angabe eines solchen $\phi \in [0, 2\pi)$ gegeben. □

10.32 Def (Definition der Exponentialfunktion durch die Euler-Formel)**.** Wir erweitern die **Exponentialfunktion** $e^x : \mathbb{R} \to \mathbb{R}$ in der Variablen $x \in \mathbb{R}$ zur Exponentialfunktion zur Funktion $e^z : \mathbb{C} \to \mathbb{C}$ in der Variablen $z \in \mathbb{C}$, indem wir

$$e^{x+\mathbf{i}y} := e^x(\cos y + \mathbf{i}\sin y)$$

für alle $x, y \in \mathbb{R}$ festlegen. (Insbesondere gilt im Fall $x = 0$ die Gleichung $e^{\mathbf{i}y} = \cos y + \mathbf{i}\sin y$ laut unserer Definition).

10.33. Jede Zahl $z \in \mathbb{C}$ besitzt eine Darstellung $z = \rho e^{i\phi}$ mit $\rho = |z| \in \mathbb{R}_{\geq 0}$ und $\phi \in \mathbb{R}$. Das ist die Umformulierung von Theorem 10.31 in den neu eingeführten Bezeichnungen.

10.34 Aufg. Zeigen Sie dass bei $z_k := \rho_k e^{\mathbf{i}\phi_k}$ mit $\rho_1, \rho_2 \in \mathbb{R}_{\geq 0}$ und $\phi_1, \phi_2 \in \mathbb{R}$ die Gleichung $z_1 z_2 = \rho_1\rho_2 e^{\mathbf{i}(\phi_1+\phi_2)}$ erfüllt ist. Benutzen Sie dafür die Formeln für $\cos(\alpha \pm \beta)$ und $\sin(\alpha \pm \beta)$.

Wenn Sie diese Formeln nicht kennen bzw. nicht gefunden haben, dann gibt es eine alternative Aufgabe: leiten Sie aus der Tatsache, dass $z_1 z_2 = \rho_1\rho_2 = \rho_1\rho_2 e^{\mathbf{i}(\phi_1+\phi_2)}$ gilt, Formeln für $\cos(\alpha \pm \beta)$ und $\sin(\alpha \pm \beta)$.

Kommentar: Formeln für $\cos(\alpha \pm \beta)$ und $\sin(\alpha \pm \beta)$ kann man im Rahmen der Linearen Algebra (IT-2) herleiten.

10.35 (Merkhilfe für trigonometrische Formeln mit Hilfe von $e^z : \mathbb{C} \to \mathbb{C}$.)**.** Es stellt sich heraus, dass sich die Rechenregeln für $e^x : \mathbb{R} \to \mathbb{R}$ direkt auf die Rechenregel in $e^z : \mathbb{C} \to \mathbb{C}$ übertragen lassen. Das Übertragen der Rechenregel im Fall einer

reellen Variablen auf den Fall einer komplexen Variablen basiert auf den (zahlreichen) trigonometrischen Formeln, die man kennt.

Umgekehrt gilt: die Rechenregel für $e^z : \mathbb{C} \to \mathbb{C}$ in Kombination mit der Euler-Formel "speichern" die trigonometrischen Formeln. Das bedeutet, dass man die trigonometrischen Formeln durch die Euler-Formel merken kann.

Stellen wir uns vor, wir haben die Formeln für $\cos 2\alpha$ und $\sin 2\alpha$ vergessen. Was wir tun können, ist Folgendes. Es gilt:

$$e^{2\alpha \mathbf{i}} = (e^{\mathbf{i}\alpha})^2.$$

Die Anwendung der Euler-Formel für die linke und rechte Seite ergibt dann

$$\cos 2\alpha + \mathbf{i} \sin 2\alpha = (\cos \alpha + \mathbf{i} \sin \alpha)^2.$$

Für die rechte Seite können wir die zweite binomische Formel anwenden. Das ergibt:

$$\cos 2\alpha + \mathbf{i} \sin 2\alpha = \cos^2 \alpha + 2 \sin \alpha \cos \alpha \mathbf{i} + \mathbf{i}^2 \sin^2 \alpha.$$

Nach einer Vereinfachung der rechten Seite erhalten wir

$$\cos 2\alpha + \mathbf{i} \sin 2\alpha = (\cos^2 \alpha - \sin^2 \alpha) + 2 \sin \alpha \cos \alpha \mathbf{i}$$

Auf diese Weise kommen wir zu den Formeln

$$\cos 2\alpha = \cos^2 \alpha - \sin^2 \alpha,$$
$$\sin 2\alpha = 2 \sin \alpha \cos \alpha.$$

10.36 Aufg. Welche trigonometrische Identitäten speichern die folgenden Formeln?

$$e^{\mathbf{i}\alpha} e^{-\mathbf{i}\alpha} = 1,$$
$$e^{\mathbf{i}(\alpha+\beta)} = e^{\mathbf{i}\alpha} e^{\mathbf{i}\beta},$$
$$e^{-\mathbf{i}\alpha} = (e^{\mathbf{i}\alpha})^{-1}.$$

10.37. A priori ist nicht klar, wieso unsere Definition der Exponentialfunktion $e^z : \mathbb{C} \to \mathbb{C}$ sinnvoll ist. Um zu verstehen, dass das die einzige richtige Weise ist, diese Funktion zu definieren, braucht man Wissen aus der Analysis (IT-3), und zwar braucht man Potenzreihen dafür.

10.38 Def. In Anlehnung an Definition 10.32 erweitern wir $\sin, \cos : \mathbb{R} \to \mathbb{R}$ zu Funktionen $\sin, \cos : \mathbb{C} \to \mathbb{C}$, indem wir

$$\cos z := \frac{e^{\mathbf{i}z} + e^{-\mathbf{i}z}}{2}$$
$$\sin z := \frac{e^{\mathbf{i}z} - e^{-\mathbf{i}z}}{2\mathbf{i}}$$

festlegen.

10.39. Die Euler-Formel ermöglicht einen Übergang von den trigonometrischen Funktionen zur Exponentialfunktion. So ein Übergang, der oft die Berechnung erleichtert, wird gerne in Physik/Elektrotechnik benutzt, wenn man sich mit Schwingungen und Wellen beschäftigt.

11 Asymptotische Notation

11.1 O, Ω und Θ

11.1. Bei der Analyse von Algorithmen und der Analysis redet man oft von der Größenordnung von Funktionen. Eine praktische Ausdrucksweise dafür ist die sogenannte asymptotische Notation.

11.2 Def (O-Notation)**.** Seien $f, g : \mathbb{N} \to \mathbb{R}$ Funktionen. Man schreibt $f(n) = O(g(n))$, wenn eine Konstante $c > 0$ und ein $n_0 \in \mathbb{N}$ existiert, so dass $|f(n)| \leq c|g(n)|$ für alle $n \geq n_0$ gilt.

11.3. Die Bezeichnung $f(n) = O(g(n))$ steht für "$f(n)$ hat die Größenordnung höchstens $g(n)$ bis auf eine mutliplikative Konstante" und man sagt „$f(n)$ ist in Groß-O von $g(n)$". Die Schreibweise $f(n) = O(g(n))$ ist streng genommen nicht ganz korrekt, in der Literatur aber sehr verbreitet. Die korrekte Schreibweise wäre $f(n) \in O(g(n))$, d.h., $f(n)$ liegt in der Menge aller Funktionen der Größenordnung höchstens $g(n)$. In der Literatur verwendet man oft $O(g(n))$ als eine Schreibweise für eine anonyme Funktion der Größenordnung höchstens $g(n)$. In diesem Kurs spielen die Beträge in der Definition von $O(g(n))$ in der Regel keine Rolle, weil wir beim Anwenden der asymptotischen Notationen fast ausschließlich nichtnegative Funktionen benutzen.

11.4 Def (Ω-Notation)**.** Man schreibt $f(n) = \Omega(g(n))$, wenn eine Konstante $c > 0$ und ein $n_0 \in \mathbb{N}$ existieren, so dass $|f(n)| \geq c|g(n)|$ für alle $n \geq n_0$ gilt. In diesem Fall: Die Größenordnung von $f(n)$ ist mindestens $g(n)$, bis auf eine multiplikative Konstante und man sagt „$f(n)$ ist in Groß-Omega von $g(n)$".

11.5 Def. Man schreibt $f(n) = \Theta(g(n))$, wenn sowohl $f(n) = O(g(n))$ als auch$f(n) = \Omega(g(n))$ gelten.

11.6. In diesem Fall: Die Größenordnung von $f(n)$ ist genau $g(n)$ bis auf eine multiplikative Konstante, und man sagt „$f(n)$ ist in Groß-Theta von $g(n)$“.

11.7. Die asymptotischen Notationen $O(g(n))$, $\Omega(g(n))$ und $\Theta(g(n))$ (und ihre weiteren Varianten) werden oft auch Landau-Symbole genannt.

11.8 Bsp. Sei $f : \mathbb{N} \to \mathbb{R}$ definiert durch $f(n) := \sqrt{2n+5}-10$. Es gilt $f(n) = \Theta(\sqrt{n})$, denn einerseits ist $\sqrt{2n+5} - 10 \leq \sqrt{2n+5} \leq \sqrt{7n} = \sqrt{7}\sqrt{n}$ für alle $n \in \mathbb{N}$, woraus $f(n) = O(\sqrt{n})$ folgt. Andererseits ist $\sqrt{2n+5}-10 \geq \sqrt{n}-10 \geq \frac{1}{2}\sqrt{n}$ für alle $n \geq 400$, woraus $f(n) = \Omega(\sqrt{n})$ folgt.

11.9 Aufg. Sind die folgenden asymptotischen Abschätzungen richtig?

- $n! = O(n^n)$
- $n^n = \Omega(n!)$
- $n! = O(2^n)$
- $n^n = O(n!)$

11.10. Seien $f_1, f_2, g_1, g_2 : \mathbb{N} \to \mathbb{R}$ Funktionen, wobei g_1, g_2 nicht-negativ sind und $f_i(n) = O(g_i(n))$, für $i = 1, 2$, vorausgesetzt wird. Dann gilt

$$f_1(n) + f_2(n) = O(g_1(n) + g_2(n)) = O(\max\{g_1(n), g_2(n)\}),$$

und

$$f_1(n) \cdot f_2(n) = O(g_1(n) \cdot g_2(n)).$$

11.2 o und ω

11.11 Def. Bei $g : \mathbb{N} \to \mathbb{R}$ steht $o(f(n))$ für die Menge aller Funktionen $f : \mathbb{N} \to \mathbb{R}$ mit der Eigenschaft, dass für jedes $c > 0$ ein $n_0 \in \mathbb{N}$ existiert derart, dass $|f(n)| \leq c|g(n)|$ für alle $n \in \mathbb{N}$ mit $n \geq n_0$ erfüllt ist. In der Literatur schreibt man oft $f(n) = o(g(n))$ an der Stelle von $f(n) \in o(g(n))$.

11.12 Def. Die Bezeichnung $\omega(g(n))$ steht für die Menge aller Funktionen $f : \mathbb{N} \to \mathbb{R}$, für welche für alle $c > 0$ ein $n_0 \in \mathbb{N}$ existiert derart, dass $|f(n)| \geq c|g(n)|$ für alle $n \in \mathbb{N}$ mit $n \geq n_0$ erfüllt ist.

Kapitel II

Kombinatorik

1 Basics der Kombinatorik

1.1. Die Haupfrage der Kombinatorik ist "Wie viele Elemente hat meine endliche Menge?". Etwas formaler geht es um die Formeln für die Anzahl der Elemente verschiedener endlicher Mengen, welche man in der diskreten Mathematik gerne benutzt.

1.2 Lem. Seien A, B endliche disjunkte Mengen. Dann ist $|A \cup B| = |A| + |B|$.

Beweis. Ist A oder B leer, so gilt die Formel trivialerweise: etwa bei $B = \emptyset$ gilt $|A \cup B| = |A \cup \emptyset| = |A| = |A| + 0 = |A| + |\emptyset| = |A| + |B|$.

Sonst nummerieren wir alle Elemente von A als $a_1, \ldots, a_s$ und B als $b_1, \ldots, b_t$, das heißt, A ist eine s-Elementige Menge $A = \{a_1, \ldots, a_s\}$ und B ist eine t-elementige Menge $B = \{b_1, \ldots, b_t\}$ mit $s, t \in \mathbb{N}$. Es gilt $a_i \neq a_j$ für $i \neq j$ mit $i, j \in \{1, \ldots, s\}$ und $b_i \neq b_j$ für $i \neq j$ mit $i, j \in \{1, \ldots, t\}$. Da A und B disjunkt sind gilt auch $a_i \neq a_j$ für alle $i \in \{1, \ldots, s\}$ und $j \in \{1, \ldots, t\}$. Somit ist $A \cup B = \{a_1, \ldots, a_s, b_1, \ldots, b_t\}$, sodass wir $A \cup B = \{c_1, \ldots, c_n\}$ mit $n = s + t$, $c_i = a_i$ für $i \in \{1, \ldots, s\}$ und $c_i = b_{i-s}$ für $i \in \{s+1, \ldots, n\}$ haben. Hierbei sind $c_1, \ldots, c_n$ nach der Konstruktion paarweise verschieden. Das zeigt, dass $A \cup B$ genau $n = s + t$ Elemente hat. □

1.3 Lem. Seien $A_1, \ldots, A_n$ ($n \in \mathbb{N}$) endliche paarweise disjunkte Mengen. Dann ist

$$\left| \bigcup_{i=1}^{n} A_i \right| = \sum_{i=1}^{n} |A_i|.$$

Beweis. Wir beweisen die die Formel durch Induktion über n. Die Formel ist rivial für $n = 1$, denn $\bigcup_{i=1}^{1} A_i = A_1$ und $\sum_{i=1}^{n} |A_i|$ ist $|A_1|$. Sei $n \in \mathbb{N}$ mit $n \geq 2$ gegeben

und sei die Formel im Fall von $n-1$ an der Stelle von n Mengen bereits verifiziert. Da die Mengen $A_1 \cup \cdots \cup A_{n-1}$ und A_n paarweise disjunkt sind, erhalten wir durch die Anwendung von Lemma 1.2 zu diesen beiden Mengen, dass

$$|A_1 \cup \cdots \cup A_n| = |A_1 \cup \cdots \cup A_{n-1}| + |A_n|$$

erfüllt ist. Aus der Induktionsvoraussetzung folgt, dass

$$|A_1 \cup \cdots \cup A_{n-1}| = \sum_{i=1}^{n-1} |A_i|$$

erfüllt ist. Somit ist

$$|A_1 \cup \cdots \cup A_n| = \sum_{i=1}^{n-1} |A_i| + |A_n| = \sum_{i=1}^{n} |A_i|.$$

□

1.4 Lem. Seien A und B endliche Mengen. Dann gilt

$$|A \cup B| = |A| + |B| - |A \cap B|.$$

Beweis. Wir können $A \cup B$ als dijsunkte Vereinigung von A und $B \setminus A$ darstellen. Die Anwendung von Lemma 1.2 zu A und $B \setminus A$ ergibt

$$|A \cup B| = |A \cup (B \setminus A)| = |A| + |B \setminus A|.$$

Die Menge B ist disjunkte Vereinigung von $B \setminus A$ und $A \cap B$. Die Anwendung von Lemma 1.2 zu $A \cap B$ und $B \setminus A$ ergibt

$$|B| = |B \setminus A| + |A \cap B|.$$

Aus den beiden Gleichungen, die wir auf diese Weise herleiten, folgt dann

$$|A \cup B| = |A| + |B \setminus A| = |A| + (|B| - |A \cap B|) = |A| + |B| - |A \cap B|.$$

□

1.5. Die Intuition hinter Lemma 1.4 ist: wir zählen alle Elemente in A sowie B ab. Dadurch werden die Elemente in $A \cap B$ doppelt abgezählt. Wir sollen also die Anzahl der Elemente in $A \cap B$ abbziehen, um auf die Anzahl der Elemente in $A \cup B$ zu kommen.

1.6 Bsp. In einer Klasse haben 15 Kinder eine Playstation, 10 Kinder eine Xbox und 3 Kinder die beiden genannten Geräte. Wie viele Kinder haben mindestens eines der beiden Geräte?

Sei P die Menge der Kinder, die eine Playstation haben, X die Anzahl der Kinder, die eien Xbox haben. Dann gilt:

$$|P \cup X| = |P| + |X| - |P \cap X| = 15 + 10 - 3 = 22.$$

1.7 Lem. Seien A und B endliche Menge. Dann gilt:

$$|A \times B| = |A| \cdot |B|.$$

Beweis. Ist A oder B leer, so sind die linke sowie rechte Seite der Gleichung gleich 0. Ansonsten stellen wir die Menge $A \times B$ kann als disjunkte Vereinigung $\bigcup_{a \in A} \{a\} \times B$ da. Lemma 1.3 ergibt

$$|A \times B| = \left| \bigcup_{a \in A} \{a\} \times B \right| = \sum_{a \in A} |\{a\} \times B|.$$

Für ein beliebiges festes $a \in A$ kann nun die Anzahl der Elemente in $\{a\} \times B$ bestimmt werden. Diese Anzahl ist $|B|$, da die Abbildung $f_a : B \to \{a\} \times B$ mit $f_a(b) := (a, b)$ bijektiv ist: denn hat man zwei verschiedene Elemente $b', b'' \in B$ so sind auch (a, b') und (a, b'') verschieden (Injektivität) und hat man ein beliebiges Element aus $\{a\} \times B$ fixiert, etwa (a, b) mit $b \in B$, so erhält man dieses Element als $f_a(b) = (a, b)$. □

1.8 Bsp. In einem Kinosaal hat man 10 Reihen mit 16 Plätzen pro Reihe. Jeder Platz kann also durch die Angabe der Reihe und die Nummer des Platzes in der Reihe als das Paar (r, n) mit $r \in \{1, \ldots, 10\}$ und $n \in \{1, \ldots, 16\}$ notiert werden. Die Anzahl der Plätze ist

$$|\{1, \ldots, 10\} \times \{1, \ldots, 16\}| = |\{1, \ldots, 10\}| \times |\{1, \ldots, 16\}| = 10 \cdot 16 = 160.$$

Dieses Beispiel ist genauso einfach wie unser Lemma 1.7, das die Idee hinter dem Beispiel ganz allgemein erfasst.

2 Kartesisches Produkt

2.1 Thm. *Seien $A_1, \ldots, A_n$ endliche Mengen ($n \in \mathbb{N}$). Dann ist*

$$|A_1 \times \cdots \times A_n| = \prod_{i=1}^{n} |A_i|.$$

Beweis. Wir beweisen die Gleichung durch Induktion über n. Für $n = 1$ erhalten wir eine triviale Identät. Sei die Gleichung für $n - 1$ Mengen für ein $n \in \mathbb{N}$ mit $n \geq 2$ efüllt. Dann erhält man wegen

$$A_1 \times \cdots \times A_n = A_1 \times (A_2 \times \cdots \times A_n)$$

durch die Anwendung von Lemma 1.7 die Gleichung

$$|A_1 \times \cdots \times A_n| = |A_1| \cdot |A_2 \times \cdots \times A_n|.$$

Anschließend erhalten wir aus der Induktionsvoraussetzung

$$|A_2 \times \cdots \times A_n| = \prod_{i=2}^{n} |A_i|,$$

woraus sich die gewünschte Gleichung für die Mengen $A_1, \ldots, A_n$ ergibt. □

2.2 (Identifikation in der Mathematik: das Selbe oder das Gleiche?)**.** In der Mathematik wird oft eine stillschweigende Identifikation von Objekten vorgenommen. Genauer sind zwischen manchen paaren von Mengen natürliche Bijektionen vorhanden, auf deren Basis diese Mengen identifiziert werden. Zum Beispiel kann man für eine Menge X die kartesische erste Potenz X^1 mit X identifizieren, weil man ein einelementiges Tupel (x) mit $x \in X$ mit $x \in X$ identifizieren kann. Man geht also davon aus, dass (x) das Selbe wie x ist. Beim Programmieren dagegen erfolgt eine solche Identifikation nicht immer automatisch: man soll dann Objekte verschiedener Datentypen explizit in einander konvertieren.

Im vorigen Beweis haben wir $(a_1, (a_2, \ldots, a_n))$, ein Paar, bei dem die zweite Komponente ein $(n-1)$-Tupel ist, stillschweigend mit $(a_1, \ldots, a_n)$ identifiziert.

2.3 Kor. Sei A endliche Menge und $n \in \mathbb{N}$. Dann gilt $|A^n| = |A|$.

Beweis. Die Behauptung folgt durch die Anwendung von Theorem 2.1 im Fall, dass alle Mengen $A_1, \ldots, A_n$ gleich A sind. □

3 Abbildungen

3.1 Thm. *Seien X, Y endliche Mengen. Dann gilt $|Y^X| = |Y|^{|X|}$, wobei man bei $X = Y = \emptyset$, $|Y|^{|X|} = 0^0 = 1$ setzt.*

Beweis. Im entarteten Fall $X = \emptyset$ gibt es nur eine Abbildung von $\emptyset$ nach Y. Sei $X \neq \emptyset$ und sei $n := |X|$, sodass wir alle Elemente von x als $x_1, \ldots, x_n$ indexieren können. Dann entspricht jede Abbildung $f : X \to Y$ einem n-Tupel $(f(x_1), \ldots, f(x_n)) \in Y^n$, wobei zwei verschiedene Abbildungen von X nach Y zwei verschiedene Tupel erzuegen. Umkgekehrt definiert jedes n-Tupel $(y_1, \ldots, y_n)$ die Abbildung $f : X \to Y$ mit $f(x_i) = y_i$ für alle $i \in \{1, \ldots, n\}$. Man sieht also, dass die Abbildung

$$Y^X \to Y^n,$$

die einem $f : X \to Y$ das Tupel $(f(x_1), \ldots, f(x_n))$ zuordnet, bijektiv ist. Wir erhalten mit der Verwendung von Korollar 2.3

$$|Y^X| = |Y^n| = |Y|^n = |Y|^{|X|}.$$

□

3.2 Bsp. Wieviele Möglichkeiten gibt es drei verschiedene Aufgaben unter vier Personen zu verteilen, wenn jede Aufgabe genau einer Personen zugeordent wird? Eine Zuordnung der Aufgaben den Personen ist eine Abbildung aus einer 3-elementigen Menge X von Aufgaben in die 4-elementige Menge Y der Personen. Wir zählen also die Abbildungen aus Y^X. Die Anzahl ist $|Y^X| = |Y|^{|X|} = 4^3 = 64$.

4 Injektive und bijektive Abbildungen

4.1 Def (Fakultät)**.** Für $n \in \mathbb{N}_0$ ist n **Fakultät** als

$$n! := \prod_{i=1}^{n} i$$

definiert. Insbesondere gilt $0! = 1! = 1$.

4.2 Def. Für $n, k \in \mathbb{N}_0$ definieren wir die **fallende Faktorielle** von n der Länge k als

$$n^{\underline{k}} := n \cdot \ldots \cdot (n - k + 1).$$

Man hat insbesondere $n^{\underline{0}} = 1$.

4.3 Def. Für Mengen X, Y bezeichnen wir als $\mathrm{Inj}(X, Y)$ die Menge aller injektiven und als $\mathrm{Bij}(X, Y)$ die Menge aller bijektiven Abbildungen von X nach Y.

4.4 Thm. *Seien X, Y endliche Mengen. Dann ist* $|\mathrm{Inj}(X, Y)| = |Y|^{\underline{|X|}}$.

Beweis. Wenn es eine injektive Abbildung f von X existiert Y exisitert, so gilt $|X| \leq |Y|$, denn $f(X)$ ist eine Teilmenge von Y, die genau so viele Elemente wie X hat. Das bedeutet, dass die Formel iim $|X| > |Y|$ erfüllt ist, weil in diesem Fall die linke sowie rechte Seite gleich 0 ist. Wir beweisen die Formeln im Fall $|X| \leq |Y|$ durch Induktion über $|X|$. Für $X = \emptyset$ gibt es genau eine injektive Abbildung von $X = \emptyset$ nach Y. Also gilt die Formel für $|X| = 0$. Nun betrachten wir X und Y mit $k = |X|$ und $k \leq |Y|$ und nehmen an, dass die Formel im Fall $k - 1 = |X| \leq |Y|$ bereits verifiziert wurde. Wir fixieren ein beliebiges $a \in X$. Jede injektive Abbildung f von X nach Y bildet das fixierte a auf eines der Elemente aus Y ab. Wir können also die Menge $\mathrm{Inj}(X, Y)$ als disjunkte Vereinigung

$$\mathrm{Inj}(X, Y) = \bigcup_{b \in Y} \mathrm{Inj}_{a,b}(X, Y),$$

mit

$$\mathrm{Inj}_{a,b}(X, Y) := \{f \in \mathrm{Inj}(X, Y) \,:\, f(a) = b\}\,.$$

Nach Lemma 1.3 gilt

$$|\mathrm{Inj}(X, Y)| = \sum_{b \in Y} |\mathrm{Inj}_{a,b}(X, Y)|$$

Jede Abbildung $f \in \mathrm{Inj}_{a,b}(X, Y)$ erzeugt die Abbildung $\tilde{f} : X \setminus \{a\} \to Y \setminus \{b\}$. Da f injektiv ist, ist $\tilde{f}$ ebenfalls injektiv. Auf diese Weise haben wir die Abbildung

$$f \mapsto \tilde{f}$$

von $\mathrm{Inj}_{a,b}(X, Y)$ nach $\mathrm{Inj}(X \setminus \{a\}, Y \setminus \{b\})$ erstellt. Die Abbildung f ist offensichtlich eine Bijektion, sodass man $|\mathrm{Inj}_{a,b}(X, Y)| = |\mathrm{Inj}(X \setminus \{a\}, Y \setminus \{b\})|$ hat. Nach der Induktionsvoraussetzung ist

$$|\mathrm{Inj}(X \setminus \{a\}, Y \setminus \{b\}| = |Y \setminus \{b\}|^{\underline{|X \setminus \{a\}|}} = (|Y| - 1)^{\underline{|X|-1}}.$$

Es folgt

$$\begin{aligned} |\operatorname{Inj}(X,Y)| &= \sum_{b\in Y} |\operatorname{Inj}_{a,b}(X,Y)| \\ &= \sum_{b\in Y} (|Y|-1)^{\underline{|X|-1}} \\ &= |Y| \cdot (|Y|-1)^{\underline{|X|-1}} \\ &= |Y|^{\underline{|X|}}. \end{aligned}$$

□

4.5 Kor. Seien X und Y endliche Mengen der gleichen Kardinalität n. Dann gilt $|\operatorname{Bij}(X,Y)| = n!$

Beweis. Haben endliche Mengen X und Y die gleiche Kardinalität, so gilt die Gleichheit $\operatorname{Bij}(X,Y) = \operatorname{Inj}(X,Y)$. Die Inklusion $\operatorname{Bij}(X,Y) \subseteq \operatorname{Inj}(X,Y)$ ist trivial, weil jede Abbildung nach der Definition injektiv ist. Umgekehrt: Ist $f : X \to Y$ injektiv, so hat $Y \setminus f(X)$ genau $|Y| - |f(X)| = |Y| - |X| = 0$ Elemente. Das bedeutet, $Y = f(X)$, sodass f auch surjektiv ist. Das zeigt die Inklusion $\operatorname{Inj}(X,Y) \subseteq \operatorname{Bij}(X,Y)$.

Nach dieser Bemerkung folgt die Behauptung direkt aus Theorem 4.4. □

5 Teilmengen gegebener Kardinalität

5.1 Def. Für eine Menge X xund $k \in \mathbb{N}_0$ bezeichnen wir als $\binom{X}{k}$ die Menge aller k-elementigen Teilmengen von X.

5.2 Bsp.

$$\binom{\{1,2,3,4\}}{2} = \Big\{\{1,2\},\{1,3\},\{1,4\},\{2,3\},\{2,4\},\{3,4\}\Big\}.$$

5.3 Def. Für $n,k \in \mathbb{N}_0$ wird der **Binomialkoeffizient** n **über** k als

$$\binom{n}{k} := \frac{n^{\underline{k}}}{k!} = \frac{n \cdot \ldots \cdot (n-k+1)}{k!}.$$

definiert.

5.4. Im Fall $0 \le k \le n$ und $n,k \in \mathbb{N}_0$ hat man $\binom{n}{k} = \frac{n!}{k!(n-k)!}$.

5.5 Thm. *Sei X endliche Menge und sei $k \in \mathbb{N}_0$. Dann gilt:*

$$\left|\binom{X}{k}\right| = \binom{|X|}{k}.$$

Beweis. In den Fällen $k > |X|$ werden die beiden Seiten der Formel gleich 0. Wir nehmen also $k \leq |X|$ an. Ist $k = 0$, werden die beiden Seiten der Formeln gleich 1. Wir nehmen also $0 < k \leq |X|$ an.

Wir zählen die injektiven Abbildungen von $I := \{1, \ldots, k\}$ nach X auf die folgende Weise auf. Bei jeder injektiven Abbildung $f : I \to X$ ist das Bild $f(I)$ eine k-elementige Teilmenge von X. Also kann man die injektiven Abbildungen $f : I \to X$ nach der Wahl von $f(I)$ gruppieren. Mit anderen Worten ist $\mathrm{Inj}(I, X)$ disjunkte Vereinigung

$$\mathrm{Inj}(I, X) = \bigcup_{B \in \binom{X}{k}} \{f \in \mathrm{Inj}(I, X) \,:\, f(I) = B\}\,.$$

Aus Lemma 1.3 folgt

$$|\,\mathrm{Inj}(I, X)| = \sum_{B \in \binom{X}{k}} |\,\{f \in \mathrm{Inj}(I, X) \,:\, f(I) = B\}\,|.$$

Jeder ijectiven Abbildung $f : I \to X$ mit einem vorgeschriebenen Bild B die bijektive Abbildung $\tilde{f} : I \to B$ mit $\tilde{f}(i) = f(i)$ zuordnen kann, und die Abbildung $f \mapsto \tilde{f}$ von $\{f \in \mathrm{Inj}(I, X) \,:\, f(I) = B\}$ nach $\mathrm{Bij}(I, B)$ ist bijektiv. Wir erhalten also

$$|\,\{f \in \mathrm{Inj}(I, X) \,:\, f(I) = B\}\,| = |\,\mathrm{Bij}(I, B)| = k!.$$

Das ergibt

$$|\,\mathrm{Inj}(I, X)| = \left|\binom{X}{k}\right| \cdot k!$$

Nach Theorem 4.4 hat man

$$|\,\mathrm{Inj}(I, X)| = |X|^{\underline{k}}.$$

Es folgt:

$$\left|\binom{X}{k}\right| = \frac{|\,\mathrm{Inj}(I, X)|}{k!} = \frac{|X|^{\underline{k}}}{k!} = \binom{|X|}{k!}.$$

□

5.6 (Doppeltes Abzählen)**.** Im Beweis des Theorems 5.5 haben wir injektive Abbildungen zwischen zwei festgelegten Mengen auf eine andere Weise als im Beweise des Theorems 4.4 abgezählt. Aus den beiden Weisen abzuzählen ergab sich dann im Beweis des Theorems 4.4 für die Anzahl der k-elementigen Teilmengen einer gegebenen Menge. Solchen Beweisansatz nennt man in der Kombinatorik **Doppeltes Abzählen**.

5.7 Aufg. Für alle $n, k \in \mathbb{N}_0$ gilt $\binom{n}{k} = \binom{n}{n-k}$. Verifizieren Sie das direkt arithmetisch und kombinatorisch, indem man eine Bijektion zwischen den Mengen $\binom{X}{k}$ und $\binom{X}{|X|-k}$ für eine n-elementige Teilmenge X erstellt.

6 Teilmengen

6.1 Thm. *Sei X endliche Menge. Dann gilt $|2^X| = 2^{|X|}$.*

Beweis. Es gibt verschiedene Ansätze zum Beweis dieser Formel. Zum Beispiel kann man eine natürliche Bijektion zwischen 2^X und $\{0,1\}^X$ erstellen und dann $|\{0,1\}^X| = |\{0,1\}|^{|X|} = 2^{|X|}$ nutzen (Aufgabe).

Wir präsentieren hier einen Beweis durch Induktion über $|X|$. Hat X 0 Elemente, so gilt $2^X = 2^\emptyset = \{\emptyset\}$. Somit ist $|2^X| = 1 = 2^0 = 2^{|X|}$. Sei X Mengen mit n Elementen, mit $n \geq 1$, und sei die Formel für Mengen X mit $n-1$ Elementen bereits verifiziert. Wir fixieren ein $a \in X$. Die Teilmengen A von X zerlegen sich nach den Bedingungen $a \in A$ und $a \in A$ in zwei disjunkte Mengen. Es gilt also

$$\begin{aligned} 2^X &= \{A : A \subseteq X,\ a \in A\} \dot\cup \{A : A \subseteq X,\ a \notin A\} \\ &= \{B \cup \{a\} : B \subseteq X \setminus \{a\}\} \dot\cup 2^{X\setminus\{a\}}. \end{aligned}$$

Das ergibt

$$|2^X| = |\{B \cup \{a\} : B \subseteq X \setminus \{a\}\}| + |2^{X\setminus\{a\}}|.$$

Die Abbildung $B \mapsto B \cup \{a\}$ ist eine Bijektion von $2^{X\setminus\{a\}}$ nach $\{B \cup \{a\} : B \subseteq X \setminus \{a\}\}$. Es folgt:

$$|2^X| = 2 \cdot |2^{X\setminus\{a\}}|$$

Da $X \setminus \{a\}$ eine $(n-1)$-elementige Menge ist, folgt nach der Induktionsovraussetzung $|2^{X\setminus\{a\}}| = 2^{|X\setminus\{a\}|} = 2^{|X|-1}$. Wir erhalten somit $2^{|X|} = 2 \cdot 2^{|X|-1} = 2^{|X|}$. □

7 Das Prinzip der Inklusion-Exklusion

7.1 Def. Seien A, X Mengen mit $A \subseteq X$. Dann nennen wir die Funktion $1_A : X \to \mathbb{R}$ mit

$$1_A(x) := \begin{cases} 1 & \text{für } x \in A, \\ 0 & \text{für } x \notin A \end{cases}$$

die **charakteristische Funktion** von A auf der Grundmenge X. (In der Regel ist die Wahl der Grundmenge aus dem Kontext klar, daher ist X nicht direkt in

der Bezeichnung 1_A vorhanden.)

7.2. Sind A, X endliche Menge mit $A \subseteq X$ so ist $|A| = \sum_{x \in A} 1_A(x)$. Darüber hinaus gilt $1_{X \setminus A} = 1_X - 1_A$.

7.3 Lem. Seien $A_1, \ldots, A_n$ ($n \in \mathbb{N}$) endliche Teilmengen einer endlichen Menge X. Dann gilt

$$1_{A_1 \cap \cdots \cap A_n} = 1_{A_1} \cdot \ldots \cdot 1_{A_n}.$$

Beweis. Sei $x \in X$. Liegt x im Durchschnitt der Mengen $A_1, \ldots, A_n$ so gilt $1_{A_1 \cap \cdots \cap A_n}(x) = 1$ aber auch $1_{A_i}(x) = 1$ für jedes $i \in \{1, \ldots, n\}$. Die Auswertung der Funktionen auf der linken und rechten Seite auf x ergibt somit den Wert 1¿ Ist x nicht im Durchschnitt der Mengen $A_1, \ldots, A_n$ so gilt $1_{A_1 \cap \cdot A_n}(x) = 0$ aber auch $1_{A_i}(x) = 1$ für ein $i \in \{1, \ldots, n\}$. Die Auswertung der Funktionen auf der linken rechten Seite der Formel ergibt somit den Wert 0. □

7.4 Lem. Sei $n \in \mathbb{N}$. Dann gilt für alle $t_1, \ldots, t_n \in \mathbb{R}$ die Gleichung

$$(1 - t_1) \cdot \ldots \cdot (1 - t_n) = \sum_{I \subseteq \{1, \ldots, n\}} (-1)^{|I|} \prod_{i \in I} t_i.$$

(Deutung der rechten Seite: die Summe über alle Teilmengen I von $\{1, \ldots, t\}$ und das Produkt aller t_i mit $i \in I$).

Beweis. Die Gleichung folgt durch das Ausmultiplizieren. Aus jeder der n Klammern au fer linken Seite der Formel kann beim Ausmultiplizieren unabhängig der Term 1 oder der Term $-t_i$ gewählt werden. Die Menge I kodiert also die Wahl der Terme in den Klammern durch die Angabe der Klammern, in denen man den Term $-t_i$ gewählt hat. Man hat also

$$(1 - t_1) \cdot \ldots \cdot (1 - t_n) = \sum_{I \subseteq \{1, \ldots, n\}} \prod_{i \in I} (-t_i) = \sum_{I \subseteq \{1, \ldots, n\}} (-1)^{|I|} \prod_{i \in I} t_i.$$

Einen formaleren Beweis kann man zum Beispiel durch Induktion über n führen (Aufgabe). □

7.5 Thm (Das Prinizip der Inklusion-Exklusion)**.** *Seien $A_1, \ldots, A_n$ ($n \in \mathbb{N}$)*

endliche Mengen. Dann gilt

$$|A_1 \cup \cdots \cup A_n| = \sum_{\emptyset \neq J \subseteq \{1,\ldots,n\}} (-1)^{|J|+1} \left| \bigcap_{j \in J} A_i \right|$$

Beweis. Wir setzen $X := A_1 \cup \cdots \cup A_n$ und betrachten die charakteristischen Funktionen von $A_1, \ldots, A_n$ auf der Grundmenge X. Dann gilt die Identität

$$(1 - 1_{A_1}) \cdots (1 - 1_{A_n}) = 0$$

auf der Menge x; denn für jedes $x \in X$ hat man $x \in A_i$ für ein $i \in \{1, \ldots, n\}$, woraus sich $1 - 1_{A_i}(x) = 0$ ergibt.

Aus Lemma 7.4 folgt dann

$$0 = \sum_{I \subseteq \{1,\ldots,n\}} (-1)^{|I|} \prod_{i \in I} 1_{A_i} = \sum_{i \in I} (-1)^{|I|} \, 1_{\bigcap_{i \in I} A_i},$$

wobei wir hier $\bigcup_{i \in \emptyset} A_i$ als X interpretieren. Daraus folgt:

$$\begin{aligned} 0 &= \sum_{x \in X} \sum_{i \in I} (-1)^{|I|} \, 1_{\bigcap_{i \in I} A_i}(x) \\ &= \sum_{I \subseteq \{1,\ldots,n\}} (-1)^{|I|} \sum_{x \in X} 1_{\bigcap_{i \in I} A_i}(x) \\ &= \sum_{I \subseteq \{1,\ldots,n\}} (-1)^{|I|} \left| \bigcap_{i \in I} A_i \right| \end{aligned}$$

Wir spalten in der vorigen Summer den Summanden zu $I = \emptyset$ ab und erhalten

$$0 = |X| + \sum_{\emptyset \neq I \subseteq \{1,\ldots,n\}} (-1)^{|I|} \left| \bigcap_{i \in I} A_i \right|,$$

was zur Formel in der Behauptung des Theorems äquivalent ist. □

8 Multimengen

8.1 Def. Ist X eine Menge, so ist eine **Multimenge** M über der Grundmenge X durch die Angabe der **Vielfachheitsabbildung** $\mu_M : X \to \mathbb{N}_0$ gegeben. Die Summe

$$\sum_{x \in X} \mu_M(x)$$

nennt man die Kardinalität der Multimenge M und bezeichnet diesen Wert als $|M|$. Wir nennen die Menge $\{x \in X : \mu_M(x) > 0\}$ den **Träger** von M.

Zwei Multimengen M und N nennen wir gleich, wenn sie den selben Träger T

haben, und für jedes $x \in T$ die Vielfachheiten von x in M und N übereinstimmen.

8.2 (Angabe von Multimengen durch die Aufzählung)**.** Bei der Angabe der Multimengen durch die Aufzählung benutzen wir die Bezeichnung $\{x_1, \ldots, x_n\}_{\neq}$. So ist z.B.

$$M = \{1, 1, 1, 1, 2, 2, 2, 3\}_{\neq}$$

eine Multimenge über $\{1, 2, 3\}$ mit $\mu_M(1) = 4, \mu_M(2) = 3$ und $\mu_M(3) = 1$. Wir benutzten in der Bezeichnung für Multimengen das untergestellte $\neq$, um explizit darauf hinzuweisen, dass es sich hierbei nicht um eine Menge handelt. In vielen Quellen benutzt man aber genau die gleiche Bezeichnung wie bei Mengen.

8.3 Def. Für $k \in \mathbb{N}_0$ bezeichnen wir als $\left(\!\binom{X}{k}\!\right)$ die Menge aller k-elementigen Multimengen, deren Träger in X enthalten ist.

Für $n, k \in \mathbb{N}_0$ führen wir die Bezeichnung $\left(\!\binom{n}{k}\!\right) := \binom{n+k-1}{k}$ ein.

8.4 Lem. Seien $n \in \mathbb{N}$ und $k \in \mathbb{N}_0$. Dann gilt:

$$\left|\{(z_1, \ldots, z_n) \in \mathbb{N}_0^n \,:\, z_1 + \cdots + z_n = k\}\right| = \left(\!\binom{n+k-1}{k}\!\right).$$

Beweis. Die Abbildung $(z_1, \ldots, z_n) \mapsto (y_1, \ldots, y_n) := (z_1 + 1, \ldots, z_n + 1)$ ist eine Bijektion von

$$Z := \{(z_1, \ldots, z_n) \in \mathbb{N}_0^n \,:\, z_1 + \cdots + z_n = k\}$$

nach

$$Y := \{(y_1, \ldots, y_n) \in \mathbb{N}^n \,:\, y_1 + \cdots + y_n = n + k\}.$$

Es reicht also aus, die Anzahl der Elemente in Y zu bestimmen. Eine Wahl von $(y_1, \ldots, y_n)$ lässt sich so veranschaulichen. Man zeichnet $n+k$ Punkte von links nach rechts. Zwischen diesen Punkt gibt es $n+k-1$ Lücken. Setzt man in genau $n-1$ Lücken Striche, so zerlegen sich Punkte in n. Bezeichnet man als y_i die Anzahl der Punkte in i-te Gruppe, so erhält $(y_1, \ldots, y_n) \in Y$. Umgekehrt erzeugt jedes $(y_1, \ldots, y_n) \in Y$ eine Angabe, welche der $n + k - 1$ Lücken mit Strichen gefüllt werden sollen. Daraus ergibt sich

$$|Z| = |Y| = \binom{n+k-1}{n-1} = \binom{n+k-1}{k}.$$

Die informelle Begründung mit Punkten, Lücken und Strichen kann mathematisch formal beschrieben werden. Es handelt sich um die Bijektion

$$(y_1, \ldots, y_n) \mapsto (y_1, y_1 + y_2, \ldots, y_1 + \cdots + y_{n-1})$$

von Y nach $X := \binom{\{1,\dots,n+k-1\}}{n-1}$. (Aufgabe: Überprüfen Sie, dass es tatsächlich eine Bijektion ist.) □

8.5 Thm. *Sei X endliche Menge und sei $k \in \mathbb{N}_0$. Dann gilt*

$$\left|\left(\!\binom{X}{k}\!\right)\right| = \left(\!\binom{|X|}{k}\!\right).$$

Mit anderen Worten: es gibt genau $\binom{n+k-1}{k}$ Multimengen mit k Elementen, deren Träger in X enthalten ist.

Beweis. Es reicht den Fall $|X| > 0$, $k > 0$ zu betrachten. Sei $n := |X|$. Wir nummerieren die Elemente in X und schreiben X als $\{x_1, \dots, x_n\}$. Jede Multimenge $M \in \left(\!\binom{X}{k}\!\right)$ ist eindeutig durch die Angabe der Werte $z_i := \mu_M(x_i)$ mit $i \in \{1, \dots, k\}$ gegeben. Daher folgt die Behauptung aus Lemma 8.4. □

9 Zählen und die Berechnung von Wahrscheinlichkeiten

IM AUFBAU

Kapitel III

Algorithmische und Programmiergrundlagen

1 Stellenwertsysteme

1.1. Im Computer werden alle Daten intern mittels der Symbole 0 und 1 dargestellt. Es ist klar, dass man alle Daten mit ganzen Zahlen darstellen kann (denn man kann Symbole mit ganzen Zahlen nummerieren). Um also zu verstehen, wie man die Daten mit 0 und 1 darstellt, muss vor Allem geklärt werden, wie man die ganzen Zahlen mit 0 und 1 darstellt. Dafür werden wir die Stellenwertsysteme einführen.

In der Informatik benutzt man meistens die Stellenwertsysteme zu den Basen $b \in \{2, 8, 10, 16\}$.

1.2 Aufg. Sei $b \in \mathbb{N}$ mit $b \geq 2$. Zeigen Sie, dass jede Zahl $z \in \mathbb{N}_0$ eine eindeutige Darstellung als

$$z = \sum_{i=0}^{k} z_i b^i \tag{III.1}$$

besitzt, wobei $k \in \mathbb{N}_0$, $z_0, \ldots, z_k \in \{0, \ldots, b-1\}$, und $z_k \neq 0$ für $z \neq 0$ gelten.

1.3 Def. In der vorigen Aufgaben heißen die Zahlen $z_0, \ldots, z_k$ die *Stellen* von z im Stellenwertsystem zur Basis b, und wir schreiben in diesem Fall $z_k \cdots z_0$ (zur Basis b) $= z$.

Wir nennen z_0 die niedrigste Stelle und z_k die höchste Stelle (zur Basis b) von z. Die Zahl z heißt $(k+1)$-stellige Zahl im Stellenwertsystem zur Basis b. Die Darstellung zur Basis b von negativen ganzen Zahlen erfolgt analog (mit Vorzeichen).

1.4 Bsp. Schriftliche Addition, Subtraktion, Multiplikation und Division zu einer beliebigen Basis b geht analog zur gewohnten Basis $b = 10$. Etwa

Addition zur Basis 2:

$$\begin{array}{cccc} & 1 & 0 & 1 \\ + & & 1 & 1 \\ \hline 1 & 0 & 0 & 0 \end{array}$$

1.5. Konvertierung einer Darstellung zu einer Basis b zur Basis 10. Direkt nach (III.1) oder mittels des sogenannten Horner-Schemas, das einem ermöglicht, die Anzahl der Multiplikationen im Vergleich zu reduzieren: Für $k = 2$,

$$z_2b^2 + z_1b + z_0 = b(bz_2 + z_1) + z_0,$$

für $k = 3$,

$$z_3b^3 + z_2b^2 + z_1b + z_0 = \underbrace{\underbrace{\underbrace{b\,(b\,\underbrace{(bz_3 + z_2)}_{\text{1. Runde}} + z_1)}_{\text{2. Runde}} + z_0}_{\text{3. Runde}}}$$

und so fort für $k \geq 4$.

1.6 Bsp. Die Konvertierung von der Basis 10 zu einer anderen Basis erfolgt durch iterative Division mit Rest. Konvertieren wir zum Beispiel die Zahl 46 in das System zur Basis 3. Es gilt

$$\begin{aligned} 46 &= 15 \cdot 3 + 1 = (5 \cdot 3 + 0) \cdot 3 + 1 = 5 \cdot 3^2 + 0 \cdot 3 + 1 \\ &= (1 \cdot 3 + 2) \cdot 3^2 + 0 \cdot 3 + 1 \\ &= 1 \cdot 3^3 + 2 \cdot 3^2 + 0 \cdot 3^1 + 1 \cdot 3^0. \end{aligned}$$

Das heißt:

$$46 \text{ (zur Basis 10)} = 1201 \text{ (zur Basis 3)}.$$

Das Horner-Schema zeigt sich hier in umgekehrter Form!

1.7 Bsp. Die Basis 10 benutzen Menschen, die Basis 2 die Computer, und die Basis 16 die Menschen, die maschinennah mit Computern arbeiten. Die Konvertierung zwischen der Basis 2 und der Basis 16 ist einfach, weil 16 eine Potenz von 2 ist. Die Basis 16 ermöglicht aber eine kompaktere Darstellung von Zahlen. Die Ziffern des 16er Systems (auch *Hexadezimal-System* genannt) sind die 16 Symbole $0, \ldots, 9, A, B, C, D, E, F$. Wir können diese Ziffern als Zahlen im Dezimalsystem oder Binärsystem darstellen:

Hexadezimal	Dezimal	Binär
0	0	0000
1	1	0001
2	2	0010
3	3	0011
4	4	0100
5	5	0101
6	6	0110
7	7	0111
8	8	1000
9	9	1001
A	10	1010
B	11	1011
C	12	1100
D	13	1101
E	14	1110
F	15	1111

Hier ist beispielsweise A zur Basis 16 gleich der 10 zu unserer Standardbasis 10, und F zur Basis 16 ist gleich der 15 zur Basis 10. Die Zahl BEE im Hexadezimal-System kann ins Binärsystem konvertiert werden indem man jede Ziffer durch ihre Binärdarstellung ersetzt.

$$BEE\,(\text{im Hexadezimalsystem}) = 1011\ 1110\ 1110(\text{im Binärsystem}).$$

Um sich zu vergewissern, dass das tatsächlich stimmt, kann man sich überlegen, was diese Gleichheit im Dezimalsystem bedeutet:

$$\begin{aligned}&\underbrace{(2^3+2^1+2^0)}_{B}16^2+\underbrace{(2^3+2^2+2^1)}_{E}16^1+\underbrace{(2^3+2^2+2^1)}_{E}16^0\\ =&2^{11}+2^9+2^8+2^7+2^6+2^5+2^3+2^2+2^1\end{aligned}$$

1.8. Um zu sehen, wie man im Computer Daten mit den Symbolen 0 und 1 darstellt, kann unter Linux oder Mac der hexdump-Befehl benutzt werden:

```
echo "aaabbbb" | hexdump -C
```

Ausgabe:

```
00000000  61 61 61 62 62 62 62 0a                           |aaabbbb.|
00000008
```

Hier ist 61 die hexadezimale Unicode-Kodierung des Buchstaben a, 62 die hexadezimale Unicode-Kodierung des Buchstaben b und $0a$ die hexadezimale Unicode-Kodierung von ‘neue Zeile’.

1.9 Aufg. Bei gebrochenen Zahlen erfolgt die Berechnung der Darstellung als Nachkommazahl zur Basis b genauso wie im Dezimalsystem. Berechnen Sie die Darstellung von 1/2, 1/3, 1/4, 1/5 und 1/6 als Nachkommazahl im Binärsystem.

1.10. Wenn man im Computer nicht-negative ganze Zahlen in einem n-Bit-Register speichert, so gehen im Fall eines arithmetischen Überlaufs die höheren Stellen verloren. Zum Beispiel gilt

$$11111110 + 00000010 = 100000000$$

zur Basis 2. (Im Dezimalsystem: $254 + 2 = 256$). Wenn man diese Berechnung in einem 8-Bit-Register ausführt ist das Resultat gleich 0. D.h., in einem n-bit Register werden die arithmetischen Operationen modulo 2^n durchgeführt.

2 Rechenprobleme

2.1 Def. Formal modelliert man *Rechenprobleme* als Eingabe-Rückgabe-Relationen. So etwa beschreibt die Relation

$$\big\{(a,p) \in \mathbb{N}^2 \,:\, p \text{ ist Primfaktor von } a\big\}$$

auf den natürlichen Zahlen das Rechenproblem 'bestimme einen Primfaktor der gegebenen natürlichen Zahl'. Das zugrundeliegende Rechenproblem algorithmisch zu *lösen* heißt, einen Algorithmus zu entwickeln, der zu jeder möglichen Eingabe eine korrekte Rückgabe bestimmt oder feststellt, dass keine korrekte Rückgabe existiert.

In unserem Beispiel ist für die Eingabe $a = 10$, die Zahl $p = 2$ eine korrekte Rückgabe. Die Zahl $p = 5$ passt auch, denn 5 ist ebenfalls ein Primfaktor von 10. Für $a = 1$ gibt es keine korrekte Rückgabe, so dass der Lösungsalgorithmus mit einer Meldung terminieren muss, dass es keine korrekte Rückgabe gibt.

2.2. Die Berechnung einer Abbildung/Funktion $f : X \to Y$ ist ein Spezialfall eines Rechenproblems. In diesem Fall ist $f(x)$ die eindeutige Rückgabe für die Eingabe $x \in X$ und damit die zugehörige Eingabe-Rückgabe-Relation gegeben durch

$$\{(x, f(x)) \in X \times Y \,:\, x \in X\}\,.$$

Zum Beispiel können wir das Problem der Potenzbildung als die Funktion $f : \mathbb{N}^2 \to \mathbb{N}$ mit $f(a,b) := a^b$ modellieren. In Worten: für gegebene $a, b \in \mathbb{N}$ soll a^b algorithmisch berechnet werden.

2.3 Def. Ein weiterer Spezialfall ist die Überprüfung einer Eigenschaft. Solche Probleme nennt man *Entscheidungsprobleme*. Sie können als Berechnung einer Funktion $f : X \to \{0, 1\}$ interpretiert werden. Etwa: $f : \mathbb{N} \to \{0, 1\}$ mit $f(n) = 1$, wenn n eine Primzahl ist, und $f(n) = 0$ sonst. In Worten: es soll überprüft werden, ob eine gegebene natürliche Zahl n eine Primzahl ist.

3 Variablen, Zuweisungen und Kontrollstrukturen

3.1. Um die Programmierung unabhängig von einer konkreten Programmiersprache zu diskutieren, benutzen wir den sogenannten *Pseudocode*. Pseudocode ist eine Codeskizze, die wie ein Computerprogramm aussieht, aber in der Regel menschenlesbarer und syntaktisch weniger eingeschränkt ist. Mit Pseudocode kann man Algorithmen beschreiben, ohne auf die technischen Details der Implementierung einzugehen, oder die Syntax der jeweiligen Programmiersprache beachten zu müssen.

3.2. Eine *Programm-Variable* ist ein Behälter für Werte bzw. Daten. Der Wert der Variable kann durch eine Zuweisung festgelegt werden. Man betrachte z.B. die folgenden zwei Zeilen:

1: $x := 5$
2: $x := 2 \cdot x + 4$

Hier wird in der ersten Zeile einer Variablen x der Wert 5 zugewiesen. Mit dem Symbol := bezeichnen wir im Pseudocode die Zuweisung. In der zweiten Zeile wird der Variablen x ein neuer Wert zugewiesen, wobei man sich bei der Zuweisung in der rechten Seite auf den aktuellen Wert bezieht. Damit kann man sich die rechte Seite als Eingabe und die linke Seite als Ausgabe der Zuweisung vorstellen.

3.3. Um mit den Variablen und Daten zu arbeiten benutzt man sogennante *Kontrollstrukturen*. Davon gibt es zwei Typen:

- *Verzweigungen*: `if-then`, `if-then-else`
- *Schleifen*: `for`, `while`, `repeat-until`

3.4. Als erstes Beispiel sei hier ein Code angegeben, der die Werte der Variablen x und y vertauscht, wenn am Anfang $x > y$ gilt:

1: **if** $x > y$ **:**

2: $t := x$
3: $x := y$
4: $y := t$
5: **end**

Das heißt, die drei Zuweisungen werden genau dann ausgeführt wenn beim Erreichen der Zeile 1 des Codes die Bedingung $x > y$ gilt. Die Variable t ist eine Zusatzvariable, die beim Vertauschen benutzt wird.

If-then-else ist analog aufgebaut. Im else-Teil stehen die Befehle, die ausgeführt werden, wenn die gegebene Bedingung *nicht* erfüllt ist. Die Verzweigungen lassen sich nach Belieben verschachteln um komplexere Handlungsanweisungen aufzubauen.

3.5. Ein *Array* A der Länge n ist eine Liste aus n Variablen, wobei die Variablen mit aufeinanderfolgenden ganzen Zahlen indiziert sind. In den allermeisten Programmiersprachen, einschließlich C und C++, werden die Arrays beginnend mit 0 indiziert. Bei Indizierung ab 1 (die wir im Pseudocode dieser Vorlesung benutzen) ist ein Array A der Länge n aus den Variablen $A[1], \ldots, A[n]$ zusammengesetzt, auf welche man durch die Angabe des Index i zugreifen kann. Die Variable $A[i]$ heißt die *i-te Komponente*, oder das *i-te Element* des Arrays A. Die Anzahl der Komponenten eines Arrays A wird als *Länge* von A bezeichnet und mit $\textsc{Länge}[A]$ notiert.

3.6. Wir illustrieren nun eine andere Kontrollstruktur, die *for*-Schleife, indem wir zeigen, wie man mit ihrer Hilfe die Summe der Elemente eines n-elementigen Arrays bestimmen kann.

1: $S := 0$
2: **for** $i := 1, \ldots, \textsc{Länge}[A]$ **:**
3: $S := S + A[i]$
4: **end**

3.7. Eine *while*-Schleife ist eine Kontrollstruktur, die aus dem Rumpf und der Bedingung besteht, wobei die Befehle des *Rumpfs* iterativ ausgeführt werden, solange die *Bedingung* erfüllt ist. In der *while*-Schliefe steht die Bedingung vor dem Rumpf, man sagt sie ist *kopfgesteuert*. In manchen Programmiersprachen gibt es auch *fußgesteuerte* Schleifen, wie z.B. die *repeat-until*-Schleife, bei denen die Bedingung nach dem Rumpf steht.

3.8. Nachfolgend ein Beispiel, das zeigt wie man die Komponenten eines Arrays mit Hilfe einer while-Schleife umkehren kann:

```
1: i := 1
2: j := Länge[A]
3: while i < j :
4:    A[i] und A[j] vertauschen
5:    i := i + 1  ▷ zum nächsten i
6:    j := j − 1  ▷ zum vorigen j
7: end
```

3.9. Im Pseudocode nutzen wir hier das Symbol ▷ für Kommentare, die den Zweck haben einzelne Abschnitte des Codes zu erläutern.

4 Prozeduren, Arten der Parameterübergabe und Rekursion

4.1. Eine *Prozedur* (Funktion, Unterprogramm) ist ein Code innerhalb eines Programms mit eigener Eingabe.

4.2. Stellen wir uns vor, wir müssen zur Lösung einer Rechenaufgabe immer wieder testen, ob $x \in [p, q]$ für gegebene $x, p, q \in \mathbb{Z}$ gilt. In diesem Fall lohnt es sich, eine sogenannte *Prozedur* anzulegen, welche genau diesen Test durchführt:

```
b = Ist-zwischen(x, p, q)
  if p ≤ x ≤ q oder q ≤ x ≤ p :
    b = Wahr
  end
  b = Falsch
```

Die Variablen x, p, q heißen *Eingabeparameter* der Prozedur und die Variable b heißt *Rückgabe-Variable*. In vielen modernen Programmiersprachen benutzt man für die Rückgabe keinen Variablennamen sondern den Befehl `return`. Das sieht dann so aus:

```
Ist-zwischen(x, p, q)
  if p ≤ x ≤ q oder q ≤ x ≤ p :
    return Wahr
  end
  return Falsch
```

Durch den Befehl `return` wird die Prozedur mit dem vorgegebenen Wert an dieser Stelle beendet.

Man kann auch Prozeduren ohne Rückgabe betrachten. In C++ sind es die Funktionen mit dem Rückgabetyp `void`.

4.3. In manchen Sprachen (wie z.B. in C++) stehen mehrere Arten der Parameterübergabe zur Verfügung, wie z.B. *Übergabe durch Kopie* und die *Übergabe durch Referenz*. Wenn zum Beispiel im vorigen Pseudocode x, p und q durch Kopie übergeben werden, so entstehen bei jedem Aufruf der Prozedur die drei Variablen x, p und q, welche dann entsprechend initialisiert werden. Etwa, bei der Ausführung von IST-ZWISCHEN(a, b, c) mit $x = a, p = b, q = c$.

4.4. Bei der Übergabe durch Referenz, ist der Eingabeparameter lediglich ein weiterer Name für eine Variable, die bereits existiert. Wir illustrieren dies am Beispiel vom Vertauschen in konkretem C++-Code:

```
void vertauschen(int& x,int& y) {
   int t=x;
   x=y;
   y=t;
}
int main() {
   int a=2,b=3;
   vertauschen(a,b);
   return 0;
}
```

Damit die Werte a und b in der `main`-Funktion vertauscht werden, müssen die Eingabeparameter x und y Referenzvariablen sein. In diesem Fall sind x und y zweite Namen für a bzw. b. Die Variable t ist eine *lokale* Variable der Funktion `vertauschen`. Sie entsteht bei jeder Ausführung von `vertauschen` und verschwindet nach der Terminierung dieser Funktion.

4.5. In der Beschreibung von Algorithmen im Pseudocode halten wir uns im Folgenden an die Konvention, bei der Parameterübergabe Arrays durch Referenz und einfache Datentypen durch Kopie zu übergeben.

4.6. Prozeduren, die sich selbst aufrufen, heißen *rekursiv*. Hier ein Beispiel einer Prozedur, die a^n für $a \in \mathbb{Z}$ und $n \in \mathbb{N}_0$ mittels einer Rekursion berechnet.

$p :=$ POTENZ(a, n)

if $n = 0$ **:**
  $p := 1$
else if n gerade **:**
  $q :=$ POTENZ$(a, n/2)$
  $p := q^2$
else:
  $q :=$ POTENZ$(a, (n-1)/2)$
  $p := aq^2$
end

Diese rekursive Umsetzung ist in vielen Situationen besser als die nicht-rekursive iterative Umsetzung mit $O(n)$ Iterationen.

4.7. Es kann überprüft werden, dass alles was man rekursiv umsetzt auch ohne Rekursion, etwa mit Schleifen und Arrays, umgesetzt werden kann. Dies gilt auch für das Potenzieren oben. Die rekursiven Umsetzungen sind aber manchmal leichter zu verstehen und oftmals eleganter.

5 Datentypen, Datenstrukturen, Zeiger und Verbunde

5.1. Ein *Datentyp* ist eine atomare, unstrukturierte Einheit, die sich durch das Zusammenfassen eines Wertebereichs und darauf definierter Operationen ergibt. Die geläufigsten Datentypen sind `boolean`, `char`, `byte`, `short`, `int`, `long`, `float` und `double`, und in jeder modernen Programmiersprache umgesetzt.

5.2. Eine *Datenstruktur* ist eine bestimmte Art der Organisation einer endlichen Menge von Daten des gleichen Datentyps (homogene Daten), die gewisse Funktionalitäten bereitstellt. Typische Operationen auf einer Datenstruktur sind *Abfragen*, die Information über die Menge zurückgeben, und *modifizierende Operationen*, die die Organisation der Daten innerhalb der Datenstruktur verändern.

5.3. Die typischsten Operationen auf einer Datenstruktur, die eine Menge S verwaltet, sind die folgenden:

- SUCHEN(S, k)

  Gibt ein Element aus S zurück, dass den Schlüsselwert k hat. Falls kein solches Element in S existiert wird NIL zurückgegeben.

- EINFÜGEN(S, x)

  Erweitert die Menge S um das Element x.

- LÖSCHEN(S, x)

  Entfernt das Element x aus der Menge S.

- MINIMUM(S)

  Gibt ein Element aus S zurück, das den kleinsten Schlüsselwert hat.

- MAXIMUM(S)

  Gibt ein Element aus S zurück, das den größten Schlüsselwert hat.

- NACHFOLGER(S, x)

  Gibt das Element aus S mit dem nächstgrößeren Schlüsselwert zu x zurück, falls x nicht bereits das Maximum von S ist. Ansonsten wird NIL zurückgegeben.

- VORGÄNGER(S, x)

  Gibt das Element aus S mit dem nächstkleineren Schlüsselwert zu x zurück, falls x nicht bereits das Minimum von S ist. Ansonsten wird NIL zurückgegeben.

Man beachte hier, dass die Operationen MINIMUM, MAXIMUM, NACHFOLGER und VORGÄNGER voraussetzen, dass die Menge S vollständig geordnet ist.

5.4. Beispiele von elementaren Datenstrukturen sind Arrays, Stacks, Warteschlangen und verkettete Listen. Desweiteren gibt es auch eine Vielzahl von komplexeren Datenstrukturen, wie Heaps, Suchbäume, Wörterbücher und Hashtabellen. Komplexere Daten*typen* sind oftmals implizit als Datenstruktur implementiert: *Strings* sind zum Beispiel im Wesentlichen (verkettete) Listen von Zeichen. Genauso können auch *Files* als Listen von Zeichen interpretiert werden (die Begriffe Datentyp und Datenstruktur überlappen sich also in manchen Kontexten).

5.5. Bei der konkreten Umsetzung bzw. Implementierung einer Datenstruktur werden oftmals sogenannte *Zeiger* verwendet. Das sind Adress-Variablen, d.h., eine Zeiger-Variable speichert die Adresse eines Ortes (einer anderen Variable) im Speicher des Rechners. Für Zeiger gibt es zwei Grundoperationen: ADRESSE von einem Objekt, und OBJEKT unter gegebener Adresse. In vielen Programmiersprachen haben die komplexen Objekte das Zeigerverhalten (z.B. Arrays in Python).

5.6 Bsp. Eine *einfach verkettete Liste* L ist eine Ansammlung endlich vieler Objekte $O_1, O_2, \ldots, O_k$, die durch Zeiger miteinander verbunden sind. Dabei zeigt L selbst auf das erste Objekt O_1, und jedes Objekt O_i besteht aus einem Element e_i und einem Zeiger. Für jeden Index $1 \leq i < k$ zeigt der Zeiger des Objektes O_i auf das Objekt O_{i+1} und der Zeiger von O_k zeigt auf NIL. Kurz kann man eine solche Liste auch wie folgt notieren:

$$L : e_1 \to e_2 \to \ldots \to e_k.$$

5.7. Die algorithmische Umsetzung eines Rechenproblems ist eng mit der gewählten Datenstruktur verknüpft. Seien zum Beispiel ganze Zahlen $a_1, a_2, \ldots, a_n \in \mathbb{N}$ gegeben und einmal als Array A, mit Einträgen $A[i] = a_i$, für $1 \leq i \leq n$, und einmal als einfach verkettete Liste $L : a_1 \to a_2 \to \ldots \to a_n$ organisiert. Die Aufgabe besteht nun darin, die Elemente in umgekehrter Reihenfolge, d.h., $a_n, a_{n-1}, \ldots, a_1$, auszugeben. Für die Liste L müssen wir erst Vorüberlegungen anstellen, die uns zu einem Algorithmus führen, dessen Anzahl der Schritte dieselbe Größenordnung hat, wie der intuitive Algorithmus $A[n], A[n-1], \ldots, A[1]$ auf dem Array A.

5.8. Ein weiteres wichtiges Werkzeug für die Organisation von Daten sind die sogenannten *Verbunde*, die auch Records, Klassen, oder Strukturen genannt werden. Im Gegensatz zu den Datenstrukturen, die homogene Daten organisieren, ermöglichen es die Verbunde, Daten verschiedener Datentypen (heterogene Daten) in einem „Päckchen“ zusammenzufassen. Wir können uns beispielsweise einen Verbund `Auto` denken, der durch sogennante *Attribute* wie Modell, Kennzeichen, Kilometerstand, usw. definiert wird. Ebenso kann man einen farbigen Punkt in der Ebene als den Verbund der drei Attribute, x-Komponente, y-Komponente und Farbe ansehen.

6 Random Access Machine

6.1. Die *Random Access Machine* (kurz *RAM*), oder auf deutsch, *Maschine mit wahlfreiem Zugriff*, wird unsere Idealisierung bzw. mathematische Abstraktion des realen Rechners sein. Alle Analysen und Entwürfe von Algorithmen in diesem Kurs werden im Rahmen der RAM durchgeführt.

Wir nehmen an, dass die Zellen unserer Maschine ganze Zahlen beliebiger Größe speichern können (d.h., die Bit-Größe der Speicherzellen ist unendlich). Die Speichergröße (d.h., die Anzahl der Speicherzellen) ist ebenfalls unbeschränkt (d.h., unendlich). Wir können desweiteren alle anderen Datentypen auf der Basis der ganzen Zahlen umsetzen.

Die Random Access Machine kann auch rein formal eingeführt werden. Wir betrachten hier (zunächst) allerdings eine etwas informelle Beschreibung, in der wir festlegen welche Datentypen, Operationen und Kontrollstrukturen für uns elementar sind.

Als *Grundoperationen* erlauben wir:

- Zuweisung (für ganzzahlige Datentypen)
- Addition von ganzzahligen Variablen
- Multiplikation einer ganzzahligen Variablen mit einer Konstanten
- Ganzzahlige Division einer ganzzahligen Variablen durch eine Konstante
- Zugriff zu Speicherzellen über einen Index
- Vergleichsoperationen $<$, $\leq$, $=$, $\geq$, $>$
- Kontrollstrukturen if-then-else, while, for

6.2. Wir lassen die Multiplikation von zwei ganzzahligen Variablen in unserem Modell nicht als Grundoperation zu. Denn, wenn das eine Grundoperation wäre, so hätte der folgende Algorithmus die Laufzeit $O(n)$:

$x := 2$
for $i = 1, \ldots, n$ **:**
  $x := x^2$
end

Dieser Algorithmus würde also 2^{2^n} in der Zeit $O(n)$ berechnen. Die Zahl 2^{2^n} hat allerdings $2^n + 1$ Binärstellen. Wir würden also eine Zahl exponentieller Bitgröße in linearer Zeit berechnen, was wir als unrealistisch ansehen.

Literaturverzeichnis

[Ber17] Berghammer: Mathematik für Informatiker. Grundlegende Begriffe und Strukturen. Springer Vieweg 2017

[Ber19] Berghammer: Mathematik für Informatiker. Grundlegende Begriffe und Strukturen und ihre Anwendung. Springer Vieweg 2019

[Big05] Discrete mathematics. Oxford University Press 2005

[Bri01] Mathematik für Inforamtiker. Einführung an praktischen Beispielen aus der Welt der Computer. München: Hanser 2001

[GR14] Goebbels, Rethmann. Mathematik für Informaitker: eine aus der Informatik motivierte Einführung mit zahlreichen Anwendungs- und Programmbeispielen. Springer Vieweg 2014

[KK15] Knauer, Knauer. Diskrete und algebraische Strukturen - kurz gefasst. Springer Spektrum 2015.

[KP09] Kreußler, Pfister. Mathematik für Informatiker: Algebra, Analysis, Diskrete Strukturen. Springer 2009.

[LLM21] Lehman, Leighton, Meyer. Mathematics for Computer Science. Lecture notes at MIT. `https://courses.csail.mit.edu/6.042/spring18/mcs.pdf`

[Sch12] Schubert. Mathematik für Informatiker: ausführlich erklärt mit vielen Programmbeispielen und Aufgaben.

[Ste01] Steger. Diskrete Strukturen 1. Kombinatorik, Graphentheorie, Algebra. Springer 2001

[Tit19] Peter Tittmann. Einführung in die Kombinatorik, 3. Auflage, Springer 2019