Skriptum - Relationen und Funktionen (v 1.1)

Dieses (lücken)-Skriptum begleitet die Vorlesung Diskrete Mathematik (Mathematik für Informatiker) an der Fachhochschule Vorarlberg. Es ist jedoch nicht vollständig und eignet sich daher (ohne Ergänzungen) nicht zur Vorbereitung auf die Prüfung. Beispiele, Beweise und zum Teil auch textuelle Ergänzungen werden während der Vorlesung gemeinsam erarbeiten.

Die Vorlesung folgt in weiten Teilen dem Buch Konkrete Mathematik (nicht nur) für Informatiker von Edmund Weitz [1]. Dieses Buch verfolgt den Ansatz, dass Informatiker*innen in Ihrem späteren Berufsleben mehr angewandte (konkrete) Mathematik benötigen anstelle von "reiner" Mathematik, welche Mathematik der Mathematik wegen betreibt und in der Lehre oft abstrakt und ohne nähere Betrachtung von Anwendungen bleibt. Unser Anwendungswerkzeug der Wahl: Programmieren! Sie lernen in unserer Vorlesung die mathematischen Begriffe, Methoden und vor allem deren Anwendungen, indem Sie diese in Computerprogramme "übersetzen". Wir werden dafür (wie auch Herr Weitz) Python und Jupyterlab verwenden.

Das Skriptum enthält mit hoher Wahrscheinlichkeit auch Fehler (typographisch sowie inhaltlich). Bitte melden Sie jeden Fehler der Ihnen auffällt. Ihre Kommilitonen (und natürlich auch ich) werden es Ihnen danken!

Relationen setzen mehrere Mengen miteinander in Beziehung.

Eine Relation zwischen zwei Mengen M und N ist eine Teilmenge des Produkts $M \times N$.

 \Rightarrow

Für $x \in M$ und $y \in N$ sagt man

"x steht (bezüglich R) in Relation zu y "

falls $(x, y) \in R$ und schreibt dafür xRy.

 $x\neg Ry$ bedeutet hingegen, daß "x nicht in Relation zu y steht".

Ist M = N nennt man R eine Relation auf M.

Relationen werden für unterschiedliche Dinge verwendet:

▷ Vergleich von Zahlen:

▶ Abbilden einer Menge auf eine andere:

 $\,\vartriangleright\,$ Zusammenfassen von Elementen einer Menge nach bestimmten Kriterien zu Klassen (== Äquivalenzrelationen \to später)

Beispiele von Relationen:

- 1) Menge M der eingeschriebenen FHV-Studierenden, N Menge aller Vorlesungen. Dann ist
 - Relation zwischen M und N welche alle geordneten Tupel mit Studierenden und ihren "eingeschriebenen" Vorlesungen enthält.
- 2) $A = \{Anton, Jakob, Katja\} = "Menge an Software-Entwickler*innen einer Firma". <math>B = \{C\#, Java, Python\} = "Menge der eingesetzten Programmiersprachen". Anton beherrscht Java, Katja C# und Java, Jakob Python. Dann gilt$

- eine Relation von der man "ablesen" kann, welchen Mitarbeiter man für welche Programmiersprache einsetzen kann.
- 3) Sei $A = \{\text{Eier}, \text{Milch}, \text{Honig}\}, B = \{\text{Huhn}, \text{Kuh}, \text{Biene}\}, \text{dann ist}$

die Relation "erzeugt von".

4) Sei $A = \{1, 2\}, B = \{1, 2, 3\}, dann ist$

eine Relation und es gilt

<u>Funktionen</u> (oder Abbildungen) sind Relationen zwischen zwei Mengen M und N, bezüglich der jedes Element $x \in M$ zu genau einem Element $y \in N$ in Relation steht.

Das kennen sie schon aus der Schule!
Jedoch ist diese übliche Schreibweise leider etwas "ungenau".
Besser wäre
da f ja eine Relation (und somit eine Menge von geordneten Paaren) ist.
Und formal ganz korrekt und eindeutig ist
f(x) ist dann ein Funktionswert und eben keine Funktion!
z.B. ist
f hingegen die Funktion selbst in der alle Information steckt.

Funktionen kann man mit Hilfe von	Mengendiagrammen	visualisieren:
-----------------------------------	------------------	----------------

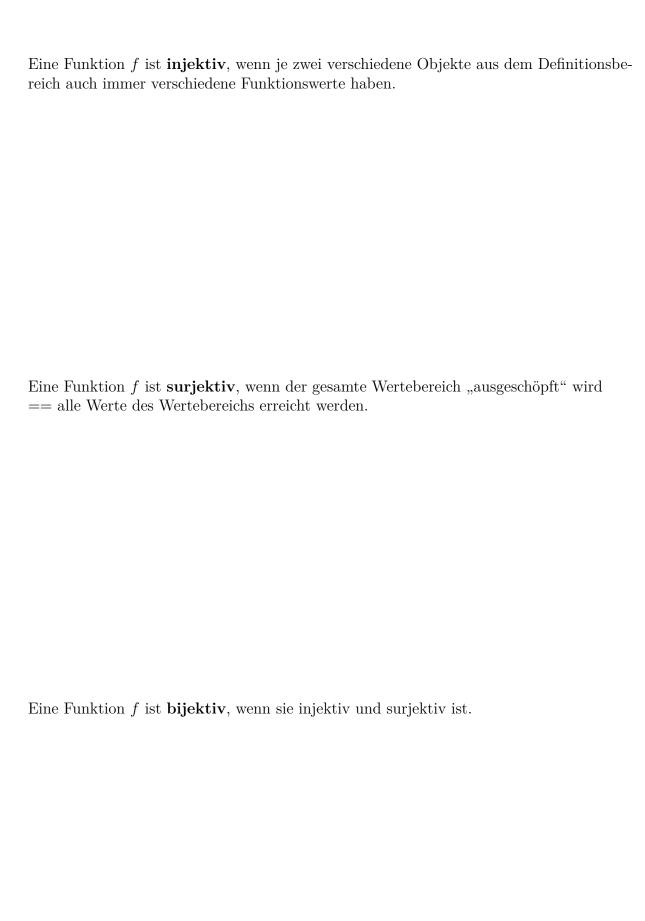
z.B.

Funktionen sind also ${\bf rechtseindeutig}$:

Es gibt immer nur ein Paar (2-Tupel) (x,y) pro $x \in M$

== es darf keine verschiedenen Paare mit identischer ersten Komponente geben.

PYTHON: functions



Injektive Funktionen können invertiert (umgekehrt) werden. == die Werte jedes Tupels werden "vertauscht".
Aus
kann man so
bilden, oder allgemein:
wobei f^{-1} Umkehrfunktion von f heißt.
Nicht injektive Funktionen sind nicht invertierbar, da das Resultat nicht mehr rechtseindeutig wäre == keine Funktion mehr ist.
Funktionen, die von $\mathbb R$ auf $\mathbb R$ abbilden kann man durch Funktionsgraphen darstellen.
z.B.: Geradengleichung:

Ųü	adratische (ыeicnung:						
Eir	nheitskreis (Kreis mit l	Mittelpunl	kt (0,0) u	nd Radius	1):		
	·		-	(, ,		,		
	ird ein Funk n nicht injel		von einer	horizonta	aler Linie 1	nehrmals ,	getroffen",	ist die l
010	и шене шје	XUIV:						

Liegt für eine Funktion eine Rechenvorschrift vor, so kann man versuchen, eine Rechenvorschrift für die Umkehrfunktion anzugeben.

Bezeichnet man den zu x gehörenden Funktionswert mit y, so wird die Funktion f durch die Gleichung y=f(x) beschrieben.

z.B.:
$$y = f(x) = 2 \cdot x - 1$$

und es muss gelten

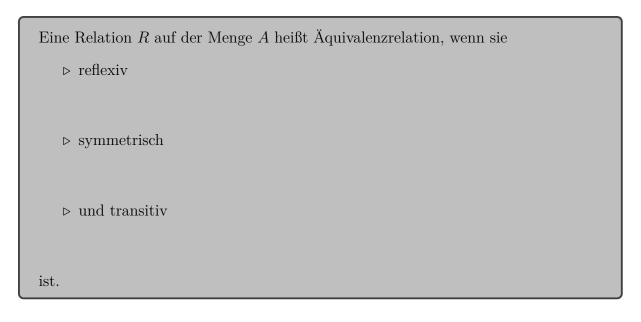
z.B.:
$$x = 2$$
 für $y = f(x) = 2 \cdot x - 1$



Da jede Menge Definitionsbereich eine lige" Funktionen definieren.	er Funktion	sein kan	n, können	wir auch	"mehrstel
z.B.:					
Ein Funktionswert ist dann z.B.					
$+$ und \cdot auf R sind somit auch zweiste	ellige Funkt	ionen:			

Äquivalenzrelation und Äquivalenzklassen

Eine Beziehung, die die Gleichwertigkeit zwischen Objekten unter bestimmten Aspekten ausdrückt, wird Äquivalenzrelation genannt. Diese werden verwendet, um Mengen in disjunkte Teilmengen zu unterteilen und finden z.B. ihre Anwendung in der Analyse großer Datenmengen.



Falls aRb mit $a \in A$ und $b \in A$ gilt (also a und b in Relation R stehen), sagt man auch "einfach", dass a (bezüglich R) zu b **äquivalent** ist.

Durch eine Äquivalenzrelation auf einer Menge A erhält man eine **Partition** oder **Klasseneinteilung** von A. Die Menge

bezeichnet man dann als Äquivalenzklassen von A bezüglich R wobei a ein beliebiges Element (Vertreter) aus dieser Klasse ist.

Beispiele:

 $\triangleright a$ wohnt im selben Ort wie b



Relationen und Datenbanken



Eine Zeile entspricht einem Element der Produktmenge
z.B. (konkrete Zeile ist Element von R_P)
Wenn wir eine zweite Tabelle Ort definieren
in der jede Zeile Element der Menge
ist, können wir Relationen auf bzw. zwischen den beiden Relationen R_P und R_O bilden und so bestimmte Daten Abfragen. Die Menge von Operationen zur Manipulation von solchen "Datenbank-Relationen" (Filtern, Verknüpfen, Aggregieren,) bilden eine relationale Algebra oder Relationenalgebra und werden z.B. von der "Structured Query Language" (SQL) implementiert.

