Nhat-Lam Luong; inf3381; 675523 – DMA Formelsammlung

Mengen A = {2, 4, 6}; **P(A)** = {{Leere Menge},{2},{4},{6},{2,4},{2,6},{4,6},{2,4,6}}

Relationen	**Funktion**
Relation auf Menge A	$R \subseteq A \times A$
Anzahl Relationen	$\lvert P(AxB)\rvert = 2\string^(\lvert AxB\rvert) = 2\string^(\lvert A\rvert*\lvert B\rvert) = 2\string^(3*2) = 2\string^6$
Reflexiv	$\forall x \in A : xRx$
Symmetrisch	$\forall x,y \in A : xRy \Rightarrow yRx$
Antisymmetrisch	$\forall x,y \in A : xRy \Rightarrow y!Rx$
Transitiv	$\forall x,y,z \in A : xRy \cap yRz \Rightarrow xRz \in R$
Äquivalenzrelation	reflexiv + symmetrisch + transitiv
Partial/Total	reflexiv/irr + antisymmetrisch + transitiv

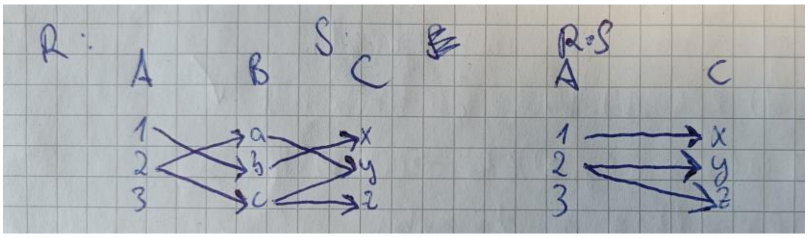

Figure - Komposition

Funktionen	**Funktion**
Injektiv	f(x) = f(y); Check: Jedes Y hat höchstens ein X-Wert
surjektiv	f(x) = y → x = ? = !f(y); Check: Alle Y werden getroffen
Bijektiv -> invertierbar	Wenn injektiv und surjektiv
Invertieren	!f(x) = swap_xy_(!f(y))
g o f; !g o g = g o !g = id(x)	g(f(x)); !g(g(x)) = g(!g(x)) = id(x)
Ungerade Wurzel	Pull Negatives
Division mit 0	0 im Nenner geht nicht, aber in Zähler schon
Modulo	**Funktion**
Paritätsbit	#1 + P are even
x mod y	x mod y = x-(Abrunden_(x/y)*y) If x < y, r = x; If x=y, r = 0; If x = y*c, r = 0
-x mod y	-x mod y = -x+Result>=$\lvert x\rvert$_(y*c) If x>y, r=x; If -x=-y, r = 0; If $\lvert x\rvert$ = y*c, r = 0
ISBN	1. SUM1_(jede 2te Nummer von rechts *3) 2. SUM2_(restliche Nummern) 3. SUM_(SUM1, SUM2) mod 10 = 0 → richtig 4. P anpassen, € 0-9
Kreditkarte VISA	1. SUM1_(jede 2te Nummer von rechts *2), Quersumme bei >9, statt 15 = 1+5 2. SUM2_(restliche Nummern) 3. SUM_(SUM1, SUM2) mod 10 = 0 → richtig 4. P anpassen, € 0-9
Abelsche Gruppe	1.) Abgeschlossenheit: a ★ b = c € M 2.) Assoziativität: a★(b★c) = (a★b)★c (Halbgroup) 3.) linksneutrales Element: e★a = a (Monoid) 4.) linksinverses Element: !a★a = e (Group) 5.) Kommutativität: a★b = b★a (if abelsche)

Matrizen	**Funktion**
A^n // != An	Verbindungsmöglichkeiten mit n Kantenschritten
Matrizenaddition = M1 + M2	Alle 1nsen von beiden Matrizen zu einer Matrix
Matrizenmultiplikation	M1 * M2 = M3 Für die einzelnen M3 Felder jeweils das dazugehörige M1 Reihenfeld und M2 Spaltenfeld nach mindestens einem gemeinsamen 1er Paar absuchen, dann ist das resultierende M3 Feld auch 1.
Graphen Algorithmen	**Funktion**
Floyd Warshall: Erreichbarkeitsmatrix	Iterationen ab 1 (A0 = Adjazenz Matrix) von Reihe und Spalte, nach gemeinsamen 1nsen schauen. bei Gew: Iterationsgewichtspaar < bisheriger Gewichtswert → ersetzen
Dijkstra: Erreichbarkeit von Startknoten aus	# | Q, S=E | A | B | C | D | ... 1. Erreichbarkeit S; kurzWeg X 2. Q-X; Erreichbarkeit X; kurzWeg Y...
Minimaler Spannbaum	Nacheinander das kürzeste Einfügen, bis alle Knots ohne Loops
Vorgängernotation	X oben, Vorgänger unten
Eulertour (Weg mit allen Kanten einmal)	**Zusammenhängend:** e1-mehr als zwei Knoten mit ungerade Grade → !Tour e2-Keine ungerade Grade → Tour e3-Zwei Knoten mit ungerade Grade → Tour **Vollständig:** Ungerade Anzahl Knoten → Tour **Gerichtet:** SUM_(eing.+ausg. Kanten) == 2-mal ungerade → Tour
Hamilton (Zyklus mit allen Knoten)	NP Vollständig; Möglich bei allen vollständigen G
Bäume	**Funktion**
Order: Pre, In, Post	1st (copy), 2nd (sort asc), last (delete)
Balance	Inorder; alle mids bilden: gerade: n/2, un: hochround
Breitensuche	Knotenlistung aller Stufen von oben nach unten.
Red Black Tree	**Funktion**
If (node == red) { node.children == black }	Red follows red → Violation Black follows Black → No Violation
Rotate_left(X) Rotate_right(X)	Left stays, but right needs adjustment Right stays, but left needs adjustment

If (Z == root) // Root and NIL == black
If (Z.t == red)
If (Z.t == black AND triangle)
If (Z.t == black AND line)
	colour_black (Z)
	recolour (ptg)
	rotate (p)
	rotate (g); recolour (pg)

Diktatur	**Funktion**
Diktatur Voraussetzung	Formel angeben
Diktatur Anfang	n = 1, L/R
Diktatur Behauptung	n = n+1, replace_(SUM_n+1, SUM_V+(n+1)); L/R