Aufgabe 1: Teildisziplinen der Informatik

Nennen Sie drei für die Informatik prägende Wissenschaftlicher(innen) und skizzieren Sie jeweiligen einen ihrer Beiträge für die Informatik mit maximal 50 Worten:

Wissenschaftler(in) 1: Alan Turing

Das von ihm entwickelte Berechenbarkeitsmodell der Turingmaschine bildet eines der Fundamente der theoretischen Informatik.

Wissenschaftler(in) 2: John von Neumann

John von Neumann entwickelte u.a. eine Computerarchitektur, die Von-Neumann-Architektur benannt, die Ähnlichkeiten mit den Entwürfen von Konrad Zuse hat, in der Daten und Programm binär codiert im selben Speicher liegen. Sie definiert eine Rechnerarchitektur aus Steuereinheit und arithmetischer Einheit sowie eine Speichereinheit. Die Befehle werden seriell abgearbeitet.

Wissenschaftler(in) 3: Noam Chomsky

Entwickelte ein System formaler Grammatiken, durch die insbesondere formale Sprachen beschrieben und erzeugt werden können. Auf der Basis formaler Grammatiken können Compiler/Übersetzer für moderne Programmiersprachen entwickelt werden.

Name:_____ Matr.-Nr: _____

Aufgabe 2: Grammatiken

Gegeben sei eine Grammatik wie folgt (ε ist Leersymbol):

G = <Startsymbol, Terminale, Nicht-Terminale, Regeln> mit

Startsymbol = $Start$

Terminale = $\{k,\ a,\ i\ \}$

Nicht-Terminale = $\{X,\ Y\ \}$

Regeln = $\{\ Start \rightarrow Y,$
$Start \rightarrow XY,$
$X \rightarrow k,$
$Y \rightarrow ai,$
$Y \rightarrow aYii\}$

Entscheiden Sie, ob folgende Wörter in der durch die Grammatik definierten Sprache enthalten sind:

1. ε ~~ja~~ / nein
2. kaaiii ja / ~~nein~~
3. ai ja / ~~nein~~
4. kaaii ~~ja~~ / nein
5. aaaiiiii ja / ~~nein~~
6. aaa ~~ja~~ / nein
7. aiai ~~ja~~ / nein
8. kai ja / ~~nein~~

Bestimmen Sie die Wörter, die durch die Grammatik akzeptiert werden in der Art $a^n b^n$:

$k^? a^n i^{2n-1}$ mit $n > 0$

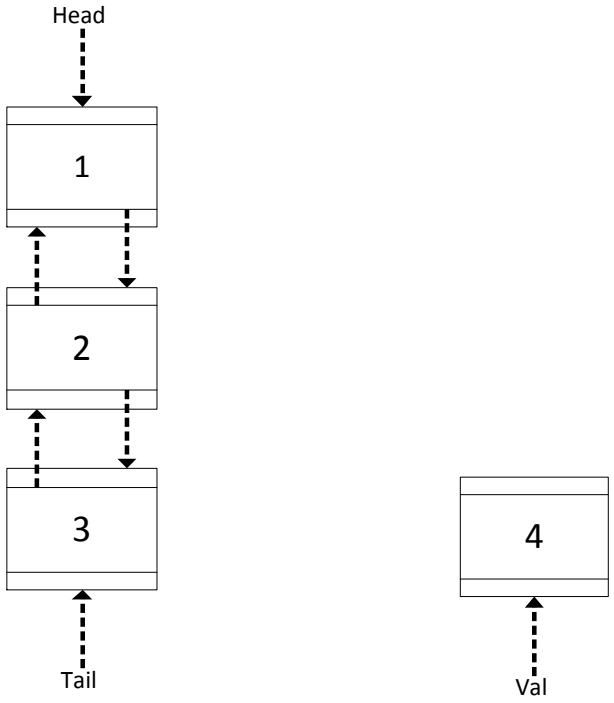
Name:_____ Matr.-Nr: _____

Aufgabe 3: Queues mit doppelt verketteten Listen

Gegeben sei eine Queue, realisiert als doppelt verkettete Liste wie unten, aufgebaut mit Strukturen der Art (Pseudocode):

ListElem { int data, pointer prev, pointer next }

als
ListElem X1, X2, X3, X4;
X1.prev = null; X1.next = adr(X2); X1.data = 1;
X2.prev = adr(X1); X2.next = adr(X3); X2.data=2;
X3.prev = adr(X2); X3.next = null; X3.data=3;
X4.prev = null; X4.next = null; X4.data = 4;
Head = adr(X1); Tail = adr(X3); Val = adr(X4);



Skizzieren Sie ein Verfahren (Pseudocode), welches das erste Element entfernt (X1) und ein neues Element (X4) hinter das letzte Element (X3) anhängt und nur von Head, Tail und Val abhängt.

Abhängen von X1: Head.next.prev = null; Head = Head.next;

Einhängen von X4: Val.prev = Tail; Tail.next = Val; Tail = Val;

Name:_____ Matr.-Nr: _____

Aufgabe 4: Prolog Unifikation

Gegeben seien jeweils 2 Prolog-Ausdrücke. Sie sollen entscheiden, ob diese beiden Ausdrücke unifizierbar sind und, falls dieses der Fall ist, den durch Substitution erzeugten Ausdruck angeben. Sollte die Unifikation misslingen, geben Sie den Grund an. Als Beispiele sind gegeben (Konstanten fangen mit kleinem, Variablen mit großem Buchstaben an):

- test (X, foo, Y) = test (M, N, M) ☺ mit test (X, foo, X).
- test (kai, X) = test1(kai, udo) ☹ weil test ≠ test1.

Hier nun die Ausdrücke:

- love(Paul,Paula) = love(bruder(paul), schwester(paula)).

☺ mit Paul = bruder(paul) und Paula = schwester(paula).

- love(paul,paula) = not(not(love(paul,paula))).

☹ love/2 ≠ not/1.

- append([1,2],[3,4],L) = append(X,Y,[1,2,3,4]).

☺ mit L = [1,2,3,4], X = [1,2], Y = [3,4].

- append([1,2],[3,4],L) = append(X,Y,[1,2,3,4,5]).

☺ mit L = [1,2,3,4,5], X = [1,2], Y = [3,4].

Name:_____ Matr.-Nr: _____

Aufgabe 5: Prolog Rekursion

Gegeben sei das Prädikat `foo`, definiert wie folgt:

```
foo(Elem1,Elem2,[Elem1|Rest]) :- member(Elem2,Rest).
foo(Elem1,Elem2,[Elem2|Rest]) :- member(Elem1,Rest).
foo(Elem1,Elem2,[_|Rest]) :- foo(Elem1,Elem2,Rest).
```

Was bedeutet dieses Prädikat (max. 20 Wörter)

Das Prädikat `foo` evaluiert zu `true`, falls das dritte Argument mit einer Liste unifizierbar ist sowie das erste Argument (`Elem1`) sowie das zweite Argument (Elem2) jeweils mit einem jeweils anderen Ausdruck unifizierbar ist, der in der Liste enthalten ist.

Gegen Sie 3 Beispiele für Aufrufe, die zu `true` evaluieren und bei Bedarf die jeweilige Variablensubstitutionen und drei Beispiele für Aufrufe, die zu `false` evaluieren:

Evaluation zu `true`:

1. `foo(1,2,[1,2,3,4,5]).`
2. `foo(2,1,[1,3,5,2,4]).`
3. `foo(X,Y,[1,2,3,4]). mit X = 1, Y = 2.`

Evaluation zu `false`:

1. `foo(1,2,3).`
2. `foo(1,2,[2,3,4]).`
3. `foo(1,1,[1,2,3,4]).`

Name:_____ Matr.-Nr: _____

Bewertung der Aufgaben

Aufgabe 1:	irgendwie zutreffende Beschreibung (maximal 3 Punkte)	=	1 Punkt
Aufgabe 2:	pro richtige Wortentscheidung	=	0.5 Punkte
	fast richtiger regulärer Ausdruck	=	0.5 Punkte
	richtige regulärer Ausdruck	=	1 Punkt
	(maximal 5 Punkte)		
Aufgabe 3:	pro richtige Codeskizze	=	3 Punkte
	pro fast richtige Codeskizze	=	2 Punkte
	pro richtige Code Idee	=	1 Punkt
	(maximal 6 Punkte)		
Aufgabe 4:	jeweils richtiges Ergebnis (maximal 4 Punkte)	=	1 Punkt
Aufgabe 5:	richtige Beschreibung	=	2 Punkte
	fast richtige Beschreibung	=	1 Punkt
	richtige Beispiele für true	=	1 Punkt
	fast richtige Beispiele für true	=	0.5 Punkte
	richtige Beispiele für false	=	1 Punkt
	fast richtige Beispiele für false	=	0.5 Punkte
	(maximal 4 Punkte)		
Maximale Punkte aller Aufgaben:		=	22 Punkte

Notengebung: Minimum(Aufrunden(Summe aller Punkte - 6),15)

Name:_____ Matr.-Nr: _____