Fr., 18.12.2015 Informatik Q2 M (Kr)

2. Klausur 13/I (Q2.1)

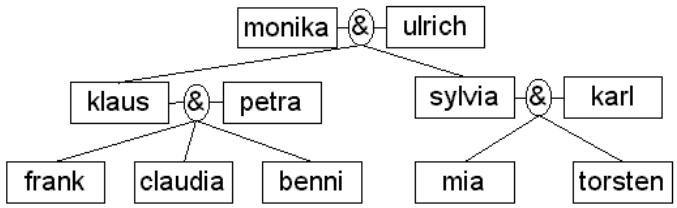
Dauer: 180 Minuten (9:15 Uhr bis 12:15 Uhr) (*195 Punkte*) Name: ___www.r-krell.de___

Hilfsmittel: Taschenrechner, Geodreieck

* *Achte auf sorgfältige Darstellung mit vollständigem, nachvollziehbarem Lösungsweg!* *
* *Kommentiere deine Java-Programme bzw. beschreibe nochmal in Deutsch die Idee deiner Prolog-Programme in den Aufgaben 2 und 3 !* *

1 Prolog (*94 Punkte*)

a) Gegeben ist nebenstehender Stammbaum. Oben steht mit Monika und Ulrich das älteste Paar. Die Paare (&) seien verheiratet und haben nur gemeinsame Kinder.



a1) [*5 P*] Schreibe 3 beispielhafte Prolog-Fakten der Form *istElterVon (Elternteil, Kind).*, und gib an, wie viele Fakten für diesen Stammbaum insgesamt nötig wären.

a2) [*7 P*] Notiere die/alle Antworte(n) auf folgende Fragen (wenn die Fakten aus a1) bekannt sind): ?- (1) *istElterVon(ulrich,klaus).* (2) *istElterVon(karl,monika)* . (3) *istElterVon (gaby,theo).* (4) *istElterVon(X,frank).* (5) *istElterVon(karl,Y).* (6) *istElterVon(mia).* Beschreibe außerdem, welche Antworten bei (7) ?- *istElterVon(X,Y).* zu erwarten sind.

a3) [*6 P*] Statt 3 Fakten der Art *verheiratet(klaus,petra).* anzugeben, kann im Beispiel *verheiratet* auch als Regel unter Verwendung von *istElterVon* definiert werden. Gib diese Regel an und notiere kurz, bei welchen (anderen) Stammbäumen bzw. Paaren diese Regel versagt. Erkläre auch, warum bei der Frage ?- *verheiratet(X,Y).* manche Paare mehrfach genannt werden.

a4) [*3 P*] Schreibe eine Prolog-Regel für *istGrosselterVon*, die z.B. auf die Frage ?- *istGrosselterVon(ulrich,benni).* mit Yes antwortet und bei ?- *istGrosselterVon(benni,mia).* No liefert.

a5) [*5 P*] Die Regel *istVorfahrVon(Alt,Jung)* soll genau dann Yes liefern, wenn *Jung* von *Alt* abstammt – egal, ob *Alt* ein Elternteil, ein Großelternteil oder (bei anderen Stammbäumen mit mehr Generationen) ein Urgroßelternteil oder noch früherer Vorfahr ist. Schreibe die Prolog-Regel.

b) [*16 P*] Für Weihnachten sollen 4 Geschenke in verschiedenfarbiges Papier eingepackt werden. Die Geschenke sind ein Teddy, ein (Spielzeug-)Auto, eine (Kino-)Karte und ein Schal. Es gibt goldenes, silbernes, blaues und rotes Geschenkpapier. Der Teddy soll aber nicht rot verpackt werden, während das Auto auf jeden Fall blau eingewickelt werden soll. Ein Prolog-Programm soll auf Anfrage erlaubte Verpackungen nennen:

b1) ?- *packe1(Teddy,Auto,Karte,Schal).* z.B. Teddy=gold, Auto=blau, Karte=rot, Schal=silber

b2) ?- *packe2(Gold,Silber,Blau,Rot).* z.B. Gold=teddy, Silber=schal, Blau=auto, Rot=karte.

Schreibe für b1) und b2) jeweils ein vollständiges Prolog-Programm, das die genannte Frage u.a. wie angegeben beantwortet.

c) [*9 P*] Für das Festtagsmenü sind als Vorspeise entweder eine Nudelsuppe (140) oder eine Tomatensuppe (160) geplant. Als Hauptspeise könnte es Weihnachtgans mit Rotkohl und Klößen („WgRK“, 1400) oder einen festlichen Tofubratling auf buntem Gemüse („TbG“, 980) oder eine Bockwurst mit Kartoffelsalat („BwKs“, 920) geben. Für die Nachspeise stehen Zimt-Eis (300) oder ein (marzipangefüllter) Bratapfel (470) zur Auswahl. Leider haben die Speisen die in Klammern angegebene Kalorienzahl. Schreibe ein Prolog-Programm aus Fakten und Regeln, das auf die Frage ?- *menue(V,H,N, K).* nacheinander alle Kombinationen mit der jeweiligen Gesamtkalorienzahl angibt, also z.B. V=tomatensuppe, H='TbG', N=zimteis, K=1440

Zusatz: Manche Menschen wollen beim Weihnachtsessen auf den Nachtisch verzichten, um

später besser bei Kuchen und Plätzchen zuschlagen zu können. Gib an, ob/wie das obige Programm so ergänzt werden kann, dass außer den Dreierkombinationen auch Speisefolgen ohne Nachtisch (aber mit Vor- und Hauptspeise) angegeben und kalorienmäßig berechnet werden.

d) In Kursen (z.B. im Kurs "Q2M8", dem 8. Mathe-Kurs in der Jahrgangsstufe Q2, der etwa der Mathematik-Leistungskurs im Mathe-Kolleg Q2M sein könnte, während "Q2M9" der Mathe-Grundkurs im Bio-Kolleg Q2Bi ist. Q2D5 sei z.B. der Deutschkurs von Q2M) sitzen viele Schüler. Jeder Kurs wird aber nur von einem Lehrer (oder 1 Lehrerin) unterrichtet.

d1) [*2 P*] Ergänze im gezeigten ER-Diagramm hier auf dem Blatt die Multiplizitäten (=Kardinalitäten) in der mc-Notation:

d2) [*8+6+5=19 P*] Notiere von Hand die benötigten drei bis fünf Tabellen dieser Schuldatenbank mit geeigneten Spaltenüberschriften (Attributen) und fülle mit jeweils einer selbst erfundenen Beispielzeile (von Schülern werde nur der Vor- und Nachname, von Kursen Angaben wie oben in Klammern und von den Lehrern Kürzel und Nachname benötigt – eventuell können oder sollten Primärschlüssel kenntlich gemacht oder hinzugefügt werden, ↑Fremdschlüssel mit Pfeil).
Schreibe anschließend alle Prolog-Fakten, um dem Prolog-System deine Tabellen bzw. die Beispiel-Inhalte mitzuteilen.
Und formuliere dann noch eine Prologregel *hatUnterrichtBei(Schuelername,Lehrername)*, die die Nachnamen aller Schüler und Lehrer ausgibt, die gemeinsame Kurse haben.

e) Euklid-Algorithmus

e1) [*5 P*] Berechne von Hand den ggT(231,30) nach dem (einfachen) Euklid-Verfahren.

e2) [*12 P*] Erläutere das nebenstehende Prolog-Programm. Erkläre insbesondere, warum in Prolog zuerst die Regel steht, die sich auf die letzte Zeile deiner Beispielrechnung bezieht – und begründe, dass diese Regel genau für die letzte Rechenzeile gilt. Erkläre außerdem, warum in Zeile 1 (A,B,B) steht. Und wie(so) funktioniert die Rekursion in der zweiten Regel? Und schließlich: Warum steht in Zeile 2 A>=B, aber in Zeile 6 nur A>B?

e3) [*2 P*] Oft kann man Prolog-Prädikate in verschiedene Richtungen benutzen (vgl. Aufg. 1a2)). Gib begründet an, ob man sich hier auch Zahlenpaare a,b mit vorgegebenem ggT erzeugen lassen kann, etwa durch ?- *euklid(A,B,3).*

e4) [*3 P*] In der Mathematik gilt ggT(231,30)=ggT(30,231). Gib an, was das Prologprogramm bei ?- *euklid(30,231,G).* liefert und ergänze ggf. das Programm so, dass immer das richtige Ergebnis heraus kommt.

*euklid.pl

```
1 euklid(A,B,B)
2 :- A >= B,
3    0 is A mod B.
4
5 euklid(A,B,GgT)
6 :- A > B,
7    R is A mod B,
8    euklid(B,R, GgT).
```

```
?- euklid(231,30,G).
solution | bindings | all bindi
yes.
G / 3
Solution: euklid(231,30,3)
```

2 Prolog-Listen (*31 Punkte*)

a) [*6 P*] Eine Prolog-Liste lässt sich als Keller verwenden. Schreibe die Operationen *istLeer(Keller), zeige1(Keller,Element), rein(AlterKeller,NeuesElement,NeuerKeller)* und *raus(AlterKeller,NeuerKeller,EntferntesElement)* in Prolog.

b) [*9 P*] Schreibe ein Prologprädikat, das auf die Frage ?- *minimum(Liste,Min).* mit einer konkreten (Zahlen-)Liste deren kleinstes Element liefert, also beispielsweise bei ?- *minimum([7,19,11,19,4],Min).* den Wert Min=4 nennt.

c) [*9 P*] Analysiere nebenstehendes Prolog-Prädikat *geheimnis*, nenne das Ergebnis von ?- *geheimnis([7,19,11,19,4],19,Ergebnis).* und beschreibe allgemein, welches Ergebnis das Prädikat bei einer beliebigen Zahlenliste und gegebenem Element *El* erzeugt.

```
geheimnis([],El,[]).
geheimnis([El|Rest],El,GRest)
:- geheimnis(Rest,El,GRest).
geheimnis([X|Rest],El,[X|GRest])
:- X\=El,
   geheimnis(Rest,El,GRest).
```

d) [*7 P*] Schreibe ein Prologprädikat *loescheEinmal*, das bei Übergabe einer (Zahlen-)Liste und eines Elements dieses Element aus der Zahlenliste löscht und die neue Liste ausgibt. Ist das Element nicht vorhanden, wird die unveränderte Liste ausgegeben. (Beispiele: ?- *loescheEinmal([7,19,11,19,4],5,NeueListe).* liefert NeueListe=[7,19,11,19,4], ?- *loescheEinmal([7,19,11,19,4],19,NeueListe).* liefert NeueListe=[7,11,19,4])

3 Sortieren (*70 Punkte*)

a) Sortiere aufsteigend von Hand die sieben Zahlen 7,19,11,19,4,23,5
   - a1) [*9 P*] mit dem Bubblesort-Verfahren, wobei du jeden Durchgang von rechts anfängst (d.h. als erstes Paar 23 und 5 vergleichst und ggf. vertauschst)
   - a2) [*19 P*] mit dem Minsort-Verfahren (mit mehreren Durchgängen von links nach rechts), wobei
     - (1) du schon innerhalb jedes Durchgangs die bisher kleinste gefundene Zahl immer sofort nach vorne tauschst
     - (2) du dir innerhalb eines Durchgangs nur die Position der kleinsten Zahl merkst bzw. ihren Index (Stelle) herausfindest und nur am Ende des Durchgangs einmal tauschst.

   Hinweis: Mache jeden Tauschvorgang von Reihungselementen (z.B. mit Pfeilen) deutlich und schreibe nach jedem Durchgang immer die vollständige Reihung einmal hin. Markiere dabei, welcher Teil der Reihung bereits endgültig ist.

b) [*15 P*] Schreibe eine Java-Methode *minsort*, die innerhalb der angedeuteten Klasse *Aufgabe3b* die Zahlen *reihe[0]* bis *reihe[n-1]* entweder nach dem Verfahren 3a2) (2) (besser, aber mehr Programmtext) oder wenigstens nach dem Verfahren 3a2) (1) (das etwas weniger Punkte bringt) sortiert. Kommentiere deinen Javatext!

```
public class Aufgabe3b
{
  int [] reihe = new reihe[1000];
  int n;  //Anzahl der Zahlen in reihe
  ...
```

c) [*9 P*] Schreibe ein Prolog-Prädikat *minsort(AlteListe,NeueListe)*, das eine Zahlenliste nach folgender Idee sortiert: Aus der Liste wird das Minimum herausgesucht, aus der Liste entfernt und dann die so verkürzte Liste sortiert. Natürlich muss man darauf achten, dass das Verfahren irgendwann endet und die Ergebnisliste richtig aufsteigend gebildet wird.
   Verwende für dein Prolog-Programm geeignete Prädikate aus 2b) und 2c)/d).

d) [*18 P*] Bestimme jeweils den Aufwand I. im besten Fall (best case) bzw. II. im schlechtesten Fall (worst case) für das Sortieren von n Zahlen nach den beiden MinSort-Varianten 3a2)(1) bzw. (2). Entwickle dazu nachvollziehbarTerme für die Anzahl V der Vergleiche bzw. die Anzahl T der Tauschoperationen von/mit Reihungselementen und gibt zum Schluss die Ordnung der Terme in der O-Notation an.
   Zusatz: Erkläre, ob/wie weit der Aufwand des Prolog-Programms aus 3c) mit einem der beiden Verfahren (welchem?) vergleichbar ist.

4 Zusatz, soweit noch Zeit (*max.12 Zusatz-Punkte*)

Betrachte das Javaprogramm auf der nächsten Seite. Vergleiche es mit Aufgabe 1c) und beurteile begründet und/oder mit Beispielen, welche Programmiersprache hier mehr Antwortmöglichkeiten bietet und vergleiche den Programmieraufwand sowie dem Aufwand bei der Ausführung des Programms in Java und Prolog, d.h. vergleiche ausgehend vom Beispiel beide Programmiersprachen.

zu Aufgabe 4: „1c) in Java“

```
public class Speise
{
 String name;
 int kalorien;

 public Speise (String bez, int kal)
 {
  name = bez;
  kalorien = kal;
 }
}
```

```
public class Festtagsmenue
{
 Speise[] vor  = {new Speise("Nudelsuppe",140), new Speise("Tomatensuppe",160)};
 Speise[] haupt= {new Speise("WgRK",1400), new Speise("TbG",980), new Speise("BwKs",920)};
 Speise[] nach = {new Speise("Zimteis",300), new Speise("Bratapfel",470)};

 public void kombiniere()
 {
  int kal;
  for (int v=0; v<2; v++)
  {
   for (int h=0; h<3; h++)
   {
    for (int n=0; n<2; n++)
    {
     kal = vor[v].kalorien + haupt[h].kalorien + nach[n].kalorien;
     System.out.println(vor[v].name+" - "+haupt[h].name+" - "+nach[n].name
      +" (zusammen "+kal+" Kalorien)");
    } // end of for n
   } // end of for h
  } // end of for v
 }
}
```

```
public class FesttagsStart
{
 public static void main (String[] s)
 {
  Festtagsmenue f = new Festtagsmenue();
  f.kombiniere();
 }
}
```