SE3	# Aufgabenblatt 6 Softwareentwicklung 3: Logikprogrammierung - WS 2016/2017 - W. Menzel Rekursive Berechnungen/rekursive Strukturen

Gesamtpunktzahl: 30 — Abgabe der Lösungen bis zum 5.12.2016

Hinweis: Bitte beachten Sie bei der Bearbeitung der Aufgaben, dass Sie

- Prädikatsdefinitionen immer übersichtlich strukturieren und ausführlich kommentieren,
- in jedem Fall ein Prädikatsschema mit Zusicherungen für die zulässigen Datentypen und den möglichen Instanziierungsvarianten für die einzelnen Argumentpositionen angeben und
- die von Ihnen durchgeführten Tests mit ihren jeweiligen Resultaten dokumentieren und ggf. diskutieren.

Aufgabe 1: Numerische Rekursion — 12 Punkte
maximale Bearbeitungszeit: 70 Minuten

Der goldene Schnitt ist das Längenverhältnis zweier Strecken a und b, für die gilt

$$\Phi = \frac{a}{b} = \frac{a+b}{a}.$$

Eine geometrische Aufteilung in diesem Verhältnis wird von vielen Menschen als besonders harmonisch empfunden und spielt insbesondere in der klassischen Architektur und Malerei ein wichtige Rolle. Der Goldene Schnitt gilt als die "irrationalste" unter den Irrationalzahlen und wird auch zur Erzeugung von Zufallszahlen (z.B. für die Kryptografie) verwendet.

Der goldene Schnitt kann durch folgende rekursive Berechnungsvorschrift approximiert werden:

$$\Phi_{n+1} = \frac{1}{\Phi_n} + 1$$

1. Definieren Sie ein Prädikat, das den goldenen Schnitt mit einer vorgegebenen Anzahl von Rekursionsschritten berechnet, z.B.

```
?- goldener_schnitt(10,Resultat).
   Resultat = 1.6180555555555556 ;
   false.
```

Ermitteln Sie einen geeigneten Startwert für die rekursive Approximation, der eine möglichst schnelle Konvergenz erlaubt.

Implementieren Sie Ihr Prädikat in zwei Varianten, wobei die Berechnung einmal beim rekursiven Abstieg und zum anderen beim rekursiven Aufstieg erfolgen soll. In welchem Fall liegt Endrekursion vor?

2. Vergleichen Sie Ihre Definitionen im Hinblick auf die Verständlichkeit und das Berechnungsverhalten.

   Den Rechenzeitbedarf eines Programmlaufs könnern Sie ermitteln, wenn Sie den Aufruf in das einstellige Prädikat `time` einbetten.

3. Der Goldenen Schnitt kann alternativ auch über den Quotienten zweier aufeinanderfolgender FIBONACCI-Zahlen berechnet werden, der für große FIBONACCI-Zahlen gegen den Goldenen Schnitt konvergiert. Berechnet man den Quotienten als Rationalzahl aus beliebig großen Integer-Zahlen, lässt sich die Güte der Approximation durch die Wahl möglichst großer FIBONACCI-Zahlen beliebig steigern. Mit der Güte der Approximation wächst die Zahl der Nachkommastellen in der Darstellung des Quotienten als Dezimalzahl an, so dass auf diese Weise gute Zufallszahlenfolgen erzeugt werden können.

   Das rekursive Prädikat

```
% fibonacci(+Rekursionstiefe,?Fibonacci-Zahl)
fibonacci(0,1).
fibonacci(1,2).
fibonacci(N,F) :-
  N > 1,
  N1 is N - 1, N2 is N - 2,
  fibonacci(N1,F1), fibonacci(N2,F2),
  F is F1 + F2.
```

   berechnet für eine gegebene Rekursionstiefe die zugehörige FIBONACCI-Zahl, ist aber leider nicht sehr effizient. Analysieren Sie die Gründe für die Ineffizienz, entwerfen Sie eine verbesserte Lösung und implementieren Sie diese.

4. Implementieren Sie ein Prädikat zur Berechnung des Goldenen Schnitts unter Verwendung des Ergebnisses aus Teilaufgabe 3. Vergleichen Sie das Berechnungsverhalten der oben angegebenen naiven Definition mit Ihrer Lösung. Untersuchen Sie Ihre Definition auf Effizienzreserven und erarbeiten Sie ggf. einen Vorschlag zur Verbesserung.

Bonus: Ersetzen Sie in Ihrer Implementation die Gleitkommadivision (Infixoperator `'/'`) durch eine Division, die Rationalzahlen ermittelt (Infixoperator `'rdiv'`).

Den numerischen Wert einer Rationalzahl können Sie sich mit Hilfe des Prädikats `format/2` als Dezimalzahl mit einer wählbaren Stellenzahl (hier im Beispiel 50) anzeigen lassen:

```
?- X is (5-3) rdiv (1+2), format('~50f',[X]).
0.66666666666666666666666666666666666666666666666667
X = 2 rdiv 3.
```

Testen Sie Ihre Implementation mit großen FIBONACCI-Zahlen. Vergleichen Sie die Genauigkeit Ihres Ergebnisses mit den Angaben auf öffentlich zugänglichen Web-Seiten. (2 Punkte)

Aufgabe 2: Stromorientierte Verarbeitung 5 Punkte
maximale Bearbeitungszeit: 30 Minuten

1. Implementieren Sie auf der Basis des Rekursionsschemas für die stromorientierte Verarbeitung eine inkrementelle Variante der in Aufgabe 1 entwickelten Prädikate für den goldenen Schnitt, die die aktuellen Approximationswerte als Folge von Lösungsalternativen über das Backtracking ermitteln, z.B.:

   ```
   ?- goldener_schnitt_incr(3,R).
      R = 2 ;
      R = 1.5 ;
      R = 1.6666666666666665 ;
      R = 1.6 ;
      false.
   ```

2. Veranschaulichen Sie sich das Konvergenzverhalten der Berechnungsvorschrift für den goldenen Schnitt grafisch. Laden Sie dazu die Datei `display.pl`. Den zeitlichen Verlauf der Reihenzerlegung können Sie sich dann durch den Aufruf des Ziels

   ```
   ?- findall(X,goldener_schnitt_incr(15,X),L),
      display('Goldener Schnitt',L).
   ```

   darstellen lassen.

   Wieviele Approximationsschritte sind erforderlich, damit bei der gegebenen Auflösung der Approximationsfehler unter die Darstellungsgenauigkeit sinkt?

Aufgabe 3: verzweigende rekursive Strukturen 13 Punkte
maximale Bearbeitungszeit: 60 Minuten

Wir betrachten rekursiv eingebettete Strukturen (Binärbäume) des Typs `s(a,b)`, `s(s(a,b),c)`, `s(a,s(b,c))`, `...` die nur aus zweistelligen Strukturen mit dem Funktor `s` bzw. aus Namen zusammengesetzt sind.

1. Definieren Sie einen Typtest für derartige Strukturen.

   Hinweis: Orientieren Sie sich am Typtest für die PEANO-Zahlen. Verwenden Sie für den Test auf das Vorliegen eines Namens das Prädikat `atom/1`.

2. Definieren Sie ein nicht-endrekursives und ein endrekursives Prädikat, das für einen Binärbaum die lokalen Einbettungstiefen auf allen Pfaden vom Spitzenknoten zu den einzelnen Blattknoten als alternative Variablenbindungen ermittelt.

   Hinweis: Orientieren Sie sich an den Prädikaten zur Umwandlung von PEANO-Zahlen. Verwenden Sie separate Klauseln für den linken und den rechten Zweig des Baumes.

3. Definieren Sie ein *nichtrekursives* Prädikat, das für einen Binärbaum die maximale Einbettungstiefe ermittelt. Sammeln Sie dazu alle lokalen Einbettungstiefen in einer Liste auf und wählen sie daraus das maximale Element. Informieren Sie sich im Handbuch über Prädikate zur Berechnung des Maximums für eine gegebene Liste von Zahlen.

4. Definieren Sie ein alternatives, *rekursives* Prädikat, das für einen Binärbaum die maximale Einbettungstiefe ermittelt.

   Hinweis: Fassen Sie die beiden rekursiven Klauseln aus einer der Lösungen für Aufgabenteil 2 zu einer einzigen Klausel zusammen und vergleichen Sie in dieser Klausel die lokalen Tiefenangaben aus dem linken und rechten Zweig des Baumes.

   Überlegen Sie sich, welche der beiden Definitionsvarianten (endrekursiv bzw. nicht-endrekursiv) auf eine einfachere Lösung führt?

5. Definieren Sie ein *rekursives* Testprädikat, das überprüft, ob ein Binärbaum balanciert ist, d.h. die lokalen Einbettungstiefen dürfen sich maximal um den Wert eins unterscheiden.

   Hinweis: Erweitern Sie die Lösung zu Aufgabenteil 4 so, dass das Prädikat zusätzlich auch die minimale Einbettungstiefe für einen (Teil-)Baum ermittelt und stellen Sie sicher, dass der Unterschied zwischen Minimum und Maximum unterhalb der geforderten Schwelle bleibt.