Hertentamen Logisch Programmeren (LIX003B05, 28 maart 2019)

Opgave 1. **De Luizenmoeder** (4 points (*punten*))

Specify whether the following Prolog terms are atoms or variables. (*Geef van de volgende Prologtermen aan of het atomen of variabelen zijn.*)

(a) `mALEDIEVE`

(b) `_Bradley`

(c) `'Juf-Ank'`

(d) `Volkert`

Opgave 2. **If all else fails...** (14 points (*punten*))

Which of the following unification attempts fail, and which succeed? (*Welke van de volgende unificatie-pogingen slagen, en welke falen*)?

(a) `?- ccc = ccc(0).`
(b) `?- D = dd.`
(c) `?- c(D) = d(C).`
(d) `?- [cc|[cc]] = [cc,cc].`
(e) `?- [dd,[]] = [dd].`
(f) `?- [c|[C]] = [c,d].`
(g) `?- [c,[c]] = [c,c].`
(h) `?- [D|[c]] = [d,c].`
(i) `?- [c,[D]] = [c,c].`
(j) `?- [e|[e|[e]]] = [e|[e,e]].`
(k) `?- [[f]] = [f].`
(l) `?- reverse([a,b,c]) = reverse([C,B,A]).`
(m) `?- c(d(C)) = c(d,C).`
(n) `?- d = D.`

Opgave 3. **That's odd!** (7 points (*punten*))

Write a recursive predicate `odd/1` that is true if and only if its argument is a non-empty Prolog list that contains an odd number of elements. (*Schrijf een recursief predikaat `odd/2` dat waar is als zijn argument een niet-lege Prologlijst is en een oneven aantal elementen bevat*):

```
?- odd([a,b,c,d,e,c,f]).
yes

?- odd([a,b,c,e,c,f]).
no

?- odd([]).
no
```

Opgave 4. **Breaking even!** (6 points (*punten*))

Write a recursive predicate `even/1` that is true if and only if its argument is a non-empty Prolog list that contains an even number of elements. (*Schrijf een recursief predikaat `even/2` dat waar is als zijn argument een niet-lege Prologlijst is en een even aantal elementen bevat*):

```
?- even([a,b,c,d,e,c,f]).
no

?- even([a,b,c,e,c,f]).
yes

?- even([]).
no
```

Opgave 5. **Mirror Words** (12 points (*punten*))

Write a predicate `mirror/1` that is true if and only if its argument is a mirror word (a word that is spelled the same backwards). Words are represented as lists of characters. Use a wrapper and a predicate with an accumulator. (*Schrijf een predikaat `mirror/1` dat waar is als zijn argument een spiegelwoord s (een woord dat van achteren naar voren hetzelfde leest als van voren naar achter)*):

```
?- mirror([m,a,d,a,m]).
yes

?- mirror([w,o,m,a,n]).
no
```

Opgave 6. **The bottle and the cork** (10 points (*punten*))

A bottle and a cork cost together 21 pounds, and the bottle costs 20 pounds more than the cork. How much do they cost each? Write a predicate that tests these constraints. The predicate requires two numbers as arguments: the first the prices of the bottle, the second the prices of the cork. (*Een fles en zijn kurk kosten bij elkaar 21 gulden, en de fles kost 20 gulden meer dan de kurk. Hoeveel kosten ze apart? Schrijf een predicaat dat dit geven test en twee getallen als argument heeft, de eerste de prijs van de fles, en de tweede de prijs van de kurk.*) For instance:

```
?- bottle_cork(20 , 1).
no

?- bottle_cork(20.5 , 0.5).
yes
```

Opgave 7. **Memory Motel** (4 points (*punten*))

Wat wordt er met *memoisation* bedoeld, en welke ingebouwde Prolog predikaten zijn hiervoor?

Opgave 8. **Around and Around** (16 points (*punten*))

Gegeven is de volgende Prolog database:

```
dneppa([X|L1],L2,[X|L3]):- dneppa(L1,L2,L3).
dneppa(L,L,[]).
```

Teken de volledige zoekbomen voor de volgende queries:

(a) `?- dneppa(A,[jagger,richards],[jones]).`
(b) `?- dneppa([watts],[jagger],[richards]).`
(c) `?- dneppa([jagger],X,[]).`

Opgave 9. **The Shuffle** (15 points (*punten*))

```
shuffle([], [], []).
shuffle(Left, Right, Merge) :-
    Left = [First | Rest],
    Merge = [First | ShortMerge],
    shuffle(Rest, Right, ShortMerge).
shuffle(Left, Right, Merge) :-
    Right = [First | Rest],
    Merge = [First | ShortMerge],
    shuffle(Left, Rest, ShortMerge).
```

Now answer the following questions:

(a) How many clauses does this program consists of?

(b) How many facts does this program consists of?

(c) How many rules does this program consists of?

(d) How many predicates are defined in this program?

(e) How many recursive predicates are defined in this program?

(f) How many tail-recursive predicates are defined in this program?

(g) What is the first answer to the query: `?- shuffle([a,b,c],[1,2,3],Shuffled).`

(h) What is the second answer to the query: `?- shuffle([a,b,c],[1,2,3],Shuffled).`

Opgave 10. **Gesneden Koek** (12 points (*punten*))

This question is about the cut predicate (!). (*Deze opgave gaat over het gebruik van het ! predikaat (de snede)*).

(a) Consider the following Prolog program (Gegeven is het volgende Prolog programma):

```
koek(X):- !, groninger_koek(X).
koek(X):- andere_koek(X).

groninger_koek(oudewijvenkoek).
groninger_koek(sucadekoek).

andere_koek(deventer_koek).
andere_koek(ontbijt_koek).
```

Welke antwoorden en in welke volgorde geeft dit programma op de query `?- koek(A).`

(b) Gegeven is het volgende Prolog programma:

```
koek(X):- groninger_koek(X), !.
koek(X):- andere_koek(X).

groninger_koek(oudewijvenkoek).
groninger_koek(sucadekoek).

andere_koek(deventer_koek).
andere_koek(ontbijt_koek).
```

Welke antwoorden en in welke volgorde geeft dit programma op de query `?- koek(B).`

(c) Gegeven is het volgende Prolog programma:

```
koek(X):- groninger_koek(X).
koek(X):- !, andere_koek(X).

groninger_koek(oudewijvenkoek).
groninger_koek(sucadekoek).

andere_koek(deventer_koek).
andere_koek(ontbijt_koek).
```

Welke antwoorden en in welke volgorde geeft dit programma op de query `?- koek(C).`

This exam has 10 questions. Total number of points: 100.
Dit tentamen bevat 10 opgaven. Puntentotaal: 100.