Tentamen Logisch Programmeren (LIX003B05, 25 januari 2018)

Opgave 1. **Unification** (16 punten)

How does Prolog answer the following queries (*Welke antwoorden geeft Prolog op de volgende queries*)?

(a) `?- [A|[]] = [hey].`
(b) `?- [een,X,drie,[]] = [een,twee,drie].`
(c) `?- [een,X,drie|[]] = [een,twee,drie].`
(d) `?- [1,2,3,4] = sort([3,1,4,2]).`
(e) `?- [een,twee|[drie,vier]] = [een,twee,drie,vier].`
(f) `?- Hey = [hey,hey].`
(g) `?- [een,twee|Drie] = [een,twee,drie].`
(h) `?- [Hey,Hey] = [hey(A),hey(b)].`
(i) `?- [een,twee|Drie] = [een,twee,3,4].`
(j) `?- [hey,hey] = HeyHey.`

Opgave 2. **Split** (12 punten)

The following Prolog database is given (*gegeven is de volgende Prolog database*):

```
split([],[],[]):- !.
split([X|L],[X|P],N):-  X > 0, !, split(L,P,N).
split([X|L],P,N):- X = 0, !, split(L,P,N).
split([X|L],P,[X|N]):- !, split(L,P,N).
```

(a) How many clauses are there in this database (*hoeveel clausules zijn er*)?
(b) How will Prolog respond to the query (*hoe reageert Prolog op de volgende query*):
`?- split([4,-5,9,0,4],X,Y).`
(c) There are four cuts in this database (*er bevinden zich vier snedes in deze database*). For each cut, describe what effect it has (*beschrijf het effect van elke snede*).

Opgave 3. **Pythagoras** (12 punten)

A Pythagorean triple consists of three positive integers a, b, and c, such that $a^2 + b^2 = c^2$ (*een Pythagorees drietal bestaat uit drie positieve gehele getallen a, b, c waarvoor geldt* $a^2 + b^2 = c^2$).

(a) Write a predicate `check_triple/3` that verifies whether three integers are a Pythagorean triple (*schrijf een predikaat `check_triple/3` dat controleert of drie getallen een Pythagorees drietal vormen*). For instance (*bijvoorbeeld*):
```
?- check_triple(3,4,5).
yes
?- check_triple(4,5,6).
no
```
(b) Write a predicate `gen_triple/3` that generates Pythagorean triples for integers smaller than 20 (*schrijf een predikaat `gen_triple/3` dat Pythagorese drietallen genereert voor gehele getallen kleiner dan 20*). For instance (*bijvoorbeeld*):
```
?- gen_triple(A,B,C).
A=3, B=4, C=5;
A=5, B=12, C=13;
etc.
```

Opgave 4. **Order** (8 punten)

Write a recursive predicate `order/1` that is true if and only if its argument is a (non-empty) ordered list of integers (in increasing order) (*Schrijf een predikaat `order/1` dat alleen waar is als zijn argument een (niet-lege) lijst van gehele getallen is die in opeenvolgende grootte gesorteerd is*). Example queries (*voorbeeldqueries*):

```
?- order([1,4,8,12]).
yes

?- order([3,5,2]).
no

?- order([]).
no

?- order([25]).
yes
```

Opgave 5. **Tail** (6 punten)

What is meant by tail-recursion, and why is it good/bad (*wat wordt met staart-recursie bedoeld en waarom is het goed/slecht*?

Opgave 6. **Semordnilap** (10 punten)

A semordnilap is a word that spells a different word in reverse. An example is "stressed", because when spelled backwards, it becomes "desserts". However, "noon" is a palindrome but not a semordnilap, because it is the same word whether spelled backward or forward. *Een semordnilap is een woord dat andersom een ander woord spelt. Een voorbeeld is "stip", want andersom gespeld is het "pits". Echter, "neven" is een palindroom en geen semordnilap omdat het andersom hetzelfde woord oplevert.*

Write a predicate `semordnilap/1` that is true if its argument is a semordnilap. Words are represented by lists of characters. Make use of the already defined predicate `word/1` which is true if its argument is an existing English or Dutch word and false otherwise. *Schrijf een predikaat `semordnilap/1` dat waar is als zijn argument andersom een ander woord spelt. Woorden woorden in lijsten van karakters weergegeven. Maak gebruik van het al gedefinieerde predikaat `word/1` dat alleen waar is als zijn argument een bestaand Engels of Nederalnds woord is.*

Opgave 7. **Variable** (4 punten)

What is an anonymous variable, how is it represented in Prolog, and for what purposes is it used (*wat is een anonieme variabele, hoe wordt die grepresenteerd in Prolog, en voor welk doeleinde worden ze gebruikt*?

Opgave 8. **Allen's Interval Algebra** (9 punten)

Given is the following Prolog database with definition of some of Allen's interval algebra (*gegeven is de volgende Prolog database met definities van enkele relaties van Allens intervalalgebra)*:

```
before(interval(A,B),interval(C,D)):- B < C.
meets(interval(A,B),interval(B,C)):- A < C.
overlap(interval(A,B),interval(C,D)):- B > C, A < C, B < D.
equal(A,A):- interval(A).
interval(interval(A,B)):- A < B.
```

(a) How many clauses does this database contain (*hoeveel clausules*)?
(b) How many rules are there in this database (*hoeveel regels*)?
(c) Which predicates are there defined in this database (*Welke predikaten zijn er in deze database gedefinieerd*)?
(d) How many recursive predicates are defined in this database (*Hoeveel recursieve predikaten zijn er in deze database gedefinieerd*)?

Opgave 9. **Magic** (14 punten)

Gegeven is de volgende Prolog database:

```
magic([X|L],L,X).
magic([X|L1],[X|L2],Y):- magic(L1,L2,Y).
```

Teken de volledige zoekbomen voor de volgende twee queries:

(a) `?- magic([sim,sala,bim],A,sala).`
(b) `?- magic([sim,sala,bim],[A,B],sala]).`

Opgave 10. **Memoisation** (9 punten)

Memoisation is an optimization technique used to speed up Prolog programs by storing the results of expensive predicate calls and returning the cached result when the same inputs occur again. (*Memoization is een techniek die gebruikt wordt om Prolog-programma's te optialiseren. Dit gebeurt door resultaten van dure berekeningen op te slaan in het geheugen en deze terug te geven als ze nog een keer gevraagd worden.*)

The following predicate computes the factorial of a number (the factorial of 0 is 1, the factorial of 5 is 5 x 4 x 3 x 2 x 1 = 120). (*Het volgende predikaat berekent de faculteit van een getal (de faculteit van 0 is 1, de faculteit van 5 is 5 x 4 x 3 x 2 x 1 = 120).*)

```
factorial(0,1).
factorial(N,F):- N > 0, M is N -1, factorial(M,G), F is G*N.
```

Optimize this Prolog program using the memoisation technique with the help of the built-in `assert/1` predicate.

Dit tentamen bevat 10 opgaven. Puntentotaal: 100. Succes!