Prof. Dr. Stephan Kleuker
Hochschule Osnabrück
Fakultät Ing.-Wissenschaften und Informatik
- Software-Entwicklung -

Logik

Sommersemester 2023
11. Aufgabenblatt

Aufgabe 18 (8 Punkte)

Laden Sie von der Veranstaltungsseite die Prolog-Datei AufgabeRechteverwaltung.pl, mit dem stark vereinfacht Rechte verwaltet werden sollen. Dabei beschreibt das Prädikat recht/3 wer, aus was (also welches Artefakt, wie ein user-Verzeichnis), wie (schreibend, lesend) zugreifen kann. Schreiben Sie folgende Prädikate und sorgen Sie dafür, dass die zugehörigen Tests erfüllt werden. Es werden jeweils Listen mit Strings ohne doppelte berechnet.

`darfUserLesenOderSchreiben(Namen):` Welche nutzenden Personen (Ergebnis: wer) dürfen das Artefakt „user“ lesen oder schreiben?
`darfLesenUndSchreiben(Namen):` Welche nutzenden Personen (Ergebnis: wer) haben bei zumindest einem Artefakt gleichzeitig Lese- und Schreibrechte?
`darfMaximalLesendZugreifen(Namen):` Welche nutzenden Personen (Ergebnis: wer) dürfen auf Artefakte ausschließlich nur lesend (oder gar nicht) zugreifen?
`artefakteOhneSchreibrecht(Artefakte):` Für welche bekannten Artefakte (Ergebnis: worauf) wurden keine Schreibrechte vergeben?
`zugriffAufAlleArtefakte(Namen):` Welche nutzenden Personen dürfen (Ergebnis: wer) auf alle bekannten Artefakte zugreifen?

Sorgen Sie dafür, dass alle Tests, auch testAll/0 laufen. Denken Sie über die Nutzung von findall/3 und das Schreiben von Hilfsprädikaten nach. Beachten Sie, dass nicht nur die Tests laufen müssen sondern Ihre Prädikate auch Lösungen berechnen können sollen.

```
?- darfUserLesenOderSchreiben(A).
A = [lala, dipsi, tinki].

?- darfLesenUndSchreiben(A).
A = [dipsi].

?- darfMaximalLesendZugreifen(A).
A = [tinki].

?- artefakteOhneSchreibrecht(A).
A = [server].

?- zugriffAufAlleArtefakte(A).
A = [tinki].
```

Aufgabe 19 (0 Punkte, freiwillig)

Realisieren Sie folgende Prädikate zu den von früher bekannten Daten, die zusammen mit den Tests als AufgabeStammbaumAdvanced.pl von der Veranstaltungsseite heruntergeladen werden können. Beachten Sie, dass nicht nur die Tests laufen müssen sondern Ihre Prädikate auch Lösungen berechnen können sollen.

a) Schreiben Sie ein zweistelliges Prädikat tante_oder_onkel/2, das true liefert, wenn die erste Person Tante oder Onkel der zweiten Person ist. Folgende Tests sollen laufen.

```
testTanteOderOnkel :-
    tante_oder_onkel('Victoria', 'Alexander'),
    tante_oder_onkel('Victoria', 'Gabriel'),
    tante_oder_onkel('Daniel', 'Gabriel'),
    tante_oder_onkel('Sofia', 'Estelle'),
    tante_oder_onkel('Oscar', 'Dipsy'),
    not(tante_oder_onkel('Victoria', 'Oscar')).
```

Prof. Dr. Stephan Kleuker
Hochschule Osnabrück
Fakultät Ing.-Wissenschaften und Informatik
- Software-Entwicklung -

b) Schreiben Sie ein zweistelliges Prädikat cousinE/2, das true liefert, wenn die erste Person Cousin oder Cousine der zweiten Person ist. Folgende Tests sollen laufen.

```
testCousinE :-
    cousinE('Estelle', 'Alexander'),
    cousinE('Alexander', 'Estelle'),
    cousinE('Estelle', 'Nicolas'),
    cousinE('Nicolas', 'Estelle'),
    not(cousinE('Estelle', 'Oscar')).
```

c) Schreiben Sie ein zweistelliges Prädikat nachfolger/2, das zu dem Namen als ersten Parameter, die korrekt sortierte Liste der direkten Nachfolgepositionen als zweiten Parameter hat. Nachfolgepositionen erhalten zunächst das älteste Kind und dessen ältestes Kind usw. Danach folgt das zweitälteste Kind und dessen Kinder usw. Folgende Tests, die diese Idee auch veranschaulichen sollen laufen. Das Entwickeln mehrerer Hilfsprädikate könnte sinnvoll sein.

```
testNachfolger :-
  nachfolger('Carl Gustav', [['Victoria',1977],['Estelle',2012]
    ,['Dipsy',2035],['Oscar',2016],['Carl Philip',1979]
    ,['Alexander',2016],['Gabriel',2017],['Madeleine',1982]
    ,['Leonore',2014],['Nicolas',2015],['Adrienne',2018]]),
  nachfolger('Victoria'
    , [['Estelle',2012],['Dipsy',2035],['Oscar',2016]]),
  nachfolger('Madeleine'
    , [['Leonore',2014],['Nicolas',2015],['Adrienne',2018]]),
  nachfolger('Oscar', []).
```