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

Logik

Sommersemester 2023
10. Aufgabenblatt Variante B

Bearbeiten Sie diese Variante des Aufgabenblattes, wenn Sie in der Nutzung von Listen noch unsicher sind.

Aufgabe 17 (9 Punkte)

Schreiben Sie folgende Prädikate in Prolog. Sie können dabei im System vorhandene Prädikate oder Prädikate aus der Vorlesung oder vorherigen Aufgabenblättern natürlich nutzen. Achten Sie darauf, dass Ihre Prädikate zur Berechnung von Ergebnissen in der Konsole und nicht nur für Überprüfungen nutzbar sind. Achten Sie weiterhin darauf, dass die hier gezeigten und von der Veranstaltungsseite erhältlichen Testprädikate als Ergebnis true liefern. Am Ende soll testAll true als Ergebnis haben.

a) Schreiben Sie ein Prädikat schnitt/3, das für zwei gegebene Listen eine Menge (als Liste) der Elemente berechnet, die in beiden Listen vorkommt. Die interaktive Nutzung soll wie folgt aussehen.

```
?- schnitt([2,2,3,4], [2,4,4,5], Erg).
Erg = [2, 4] .
```

```
testSchnitt :-
    schnitt([2,2,3,4], [2,4,4,5], Erg), % Reihenfolge egal
    length(Erg,2),
    contains(Erg, 2),
    contains(Erg, 4),
    not(contains(Erg,3)),
    not(contains(Erg,5)),
    schnitt([2,2,3,4], [2], [2]),
    schnitt([3], [2,4,3,4,8], [3]),
    schnitt([2,2,3,4], [], []),
    schnitt([], [2,4,3,4,8], []).
```

b) Schreiben Sie ein Prädikat kreuzprodukt/3, das für zwei gegebene Listen eine Liste von Paaren (ebenfalls als Liste) der Elemente berechnet, die sich aus Kombinationen der Elemente der beiden Liste ergeben.

```
?- kreuzprodukt([1,2], [3,4], Erg).
Erg = [[1, 3], [1, 4], [2, 3], [2, 4]] .
```

```
testKreuzprodukt :-
    kreuzprodukt([1,2], [3,4], Erg), % Reihenfolge egal
    length(Erg,4),
    contains(Erg, [1,3]),
    contains(Erg, [1,4]),
    contains(Erg, [2,3]),
    contains(Erg, [2,4]),
    not(contains(Erg, [1,1])),
    not(contains(Erg, [4,4])),
    kreuzprodukt([1], [2], [[1,2]]),
    kreuzprodukt([3,4], [], []),
    kreuzprodukt([], [3,4], []).
```

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

Logik

Sommersemester 2023
10. Aufgabenblatt Variante B

c) Schreiben Sie ein Prädikat listeTeilen/3, die eine gegebene Liste in zwei Teillisten mittig aufteilt. Sollte die Länge der Ausgangsliste ungerade sein, hat die erste der Teillisten ein Element mehr.

```
?- listeTeilen([1,2,2,3], L, R).
L = [1, 2],
R = [2, 3] .

?- listeTeilen([1,2,2,3,4], L, R).
L = [1, 2, 2],
R = [3, 4] .

testListeTeilen :-
    listeTeilen([1,2,2,3], [1,2], [2,3]),
    listeTeilen([1,2,2,3,4], [1,2,2], [3,4]),
    not(listeTeilen([1,2,2,3,4], [1,2], [2,3,4])),
    listeTeilen([], [], []),
    listeTeilen([1], [1], []).
```

d) Schreiben Sie ein Prädikat sort1/2, das zu einer Liste die aufsteigend geordnete Liste berechnet. Nutzen Sie den Ansatz, dass jedes Element der Ausgangsliste in eine zuvor leere Ergebnisliste schrittweise eingeordnet wird.

```
?- sort1([9,1,2,7,2,9,0,5], Erg).
Erg = [0, 1, 2, 2, 5, 7, 9, 9] .

testSort1:-
    sort1([9,1,2,7,2,9,0,5], [0,1,2,2,5,7,9,9]),
    sort1([], []),
    sort1([1], [1]),
    sort1([1,2,2,3,4,5], [1,2,2,3,4,5]),
    sort1([5,4,3,2,1], [1,2,3,4,5]),
    not(sort1([1,2,2,3], [1,2,3])),
    not(sort1([1,2,2,3], [1,2,3,3])).
```

e) Schreiben Sie ein Prädikat sort2/2, das zu einer Liste die aufsteigend geordnete Liste berechnet. Nutzen Sie den Ansatz, dass das erste Element e1 der Liste genommen und der Rest der Liste in zwei Teillisten aufgeteilt wird, wobei in der einen Teilliste die Werte kleiner als e1 und in der zweiten Teilliste die Werte größer-gleich e1 stehen. Sortieren Sie dann die beiden Teillisten nach dem gleichen Ansatz und fügen Sie am Ende die einzelnen Teile dann wieder zum Ergebnis zusammen. Die Berechnungen enden, wenn eine Liste maximal zwei Werte enthält, für die die Sortierung direkt angegeben werden kann.

```
?- sort2([9,1,2,7,2,9,0,5], Erg).
Erg = [0, 1, 2, 2, 5, 7, 9, 9] .

testSort2:-
    sort2([9,1,2,7,2,9,0,5], [0,1,2,2,5,7,9,9]),
    sort2([], []),
    sort2([1], [1]),
    sort2([1,2,2,3,4,5], [1,2,2,3,4,5]),
    sort2([5,4,3,2,1], [1,2,3,4,5]),
    not(sort2([1,2,2,3], [1,2,3])),
    not(sort2([1,2,2,3], [1,2,3,3])).
```