Fakultät Informatik/Mathematik
Lehrstuhl für Künstliche Intelligenz / Theoretische Informatik

HOCHSCHULE FÜR
TECHNIK UND WIRTSCHAFT
DRESDEN
UNIVERSITY OF APPLIED SCIENCES

Hausaufgabe

KÜNSTLICHE INTELLIGENZ

AUFGABE 20

Falk-Jonatan Strube
Bibliotheksnummer: s74053

Praktikum von
Prof. Dr. Boris Hollas

21. Juni 2017

```
% Bonus: Schritte Zeilenweise ausgeben:
printP(P) :- maplist(writeln, P), nl.

% Bonus: Suchfunktion mit formatierter Ausgabe und Infos zur benötigten Zeit:
findPath(M, S, H, R) :-
  get_time(T1),
  START = [M, [S, H], R], goal(GOAL), idDfs(START, GOAL, P), printP(P),
  get_time(T2),
  DeltaT is round((T2- T1)*1000)/1000,
  write('Weg in '), write(DeltaT), write(' s gefunden.\n'), nl, !.

:- [idDfs].

% Suchfunktion mit unformatierter Ausgabe:
findPath(M, S, H, R, P) :- START = [M, [S, H], R], goal(GOAL), idDfs(START, GOAL, P).

% Liste aller Wege (Wegraster: (0,0) ist oben links und (7,5) ist unten rechts):
weg0(P1, P2) :- member((P1, P2), [
  % Horizontale Wege (Zeilenweise):
  ((0,0), (1,0)),
  ((1,0), (2,0)),
  ((3,0), (4,0)),
  ((1,1), (2,1)),
  ((2,1), (3,1)),
  ((4,1), (5,1)),
  ((5,1), (6,1)),
  ((6,1), (7,1)),
  ((2,2), (3,2)),
  ((3,2), (4,2)),
  ((6,2), (7,2)),
  ((3,3), (4,3)),
  ((5,3), (6,3)),
  ((6,3), (7,3)),
  ((0,4), (1,4)),
  ((1,4), (2,4)),
  ((2,4), (3,4)),
  ((5,4), (6,4)),
  ((3,5), (4,5)),
  ((4,5), (5,5)),
  ((5,5), (6,5)),
  % Vertikale Wege (Zeilenweise):
  ((0,0), (0,1)),
  ((1,0), (1,1)),
  ((4,0), (4,1)),
  ((5,0), (5,1)),
  ((6,0), (6,1)),
  ((7,0), (7,1)),
```

```
48    ((1,1), (1,2)),
49    ((2,1), (2,2)),
50    ((6,1), (6,2)),
51    ((0,2), (0,3)),
52    ((2,2), (2,3)),
53    ((3,2), (3,3)),
54    ((5,2), (5,3)),
55    ((6,2), (6,3)),
56    ((0,3), (0,4)),
57    ((1,3), (1,4)),
58    ((3,3), (3,4)),
59    ((5,3), (5,4)),
60    ((7,3), (7,4)),
61    ((0,4), (0,5)),
62    ((1,4), (1,5)),
63    ((2,4), (2,5)),
64    ((3,4), (3,5)),
65    ((5,4), (5,5)),
66    ((7,4), (7,5))
67  ]).
68
69  % Überprüfe, ob ein Weg zwischen benachbarten Punkten 1 und 2
        existiert:
70  weg(P1, P2) :- weg0(P1, P2); weg0(P2, P1).
71
72  % Gültige Übergänge:
73  adj([M1, [], R1], [M1, [], R2]) :- member(M1,R1), delete(R1,M1,R2).
        % Rose schneiden
74  adj([M1, [],  R], [M2, [],  R]) :- weg(M1, M2).           % Mit
        Werkzeug bewegen
75  adj([M1, W1,  R], [M1, W2,  R]) :- member(M1,W1), delete(W1,M1,W2).
        % Werkzeug aufheben
76  adj([M1,  W,  R], [M2,  W,  R]) :- W \== [], weg(M1, M2).     %
        Ohne alle Werkzeuge bewegen
77
78  % Das Ziel: keine Rosen mehr zu schneiden!
79  goal([_,  _, []]).
80
81  % Zum Testen der Funktion:
82  % Einfach: findPath((0,0), (0,1), (1,0), [(1,1), (3,1)]).
83  % Mittel: findPath((0,0), (0,1), (1,0), [(7,5)]).
84  % Schwer: findPath((0,0), (7,5), (0,5), [(7,0), (0,0)]).
85  % Schwerer: findPath((0,0), (7,5), (0,5), [(7,0), (0,0), (3,5),
        (7,3), (2,0), (1,1)]).
```