Source Code

File Edit Browse Compile Prolog Pce Help

kelompok3_level1.pl

```
posisi(kiri).
posisi(kanan).
periksa((Orang, Domba, Serigala, Wortol)) :-
    posisi(Orang),
    posisi(Domba),
    posisi(Serigala),
    posisi(Wortol),
    Orang == Domba.
periksa((Orang, Domba, Serigala, Wortol)) :-
    posisi(Orang),
    posisi(Domba),
    posisi(Serigala),
    posisi(Wortol),
    Orang == Wortol,
    Domba \= Serigala.
periksa((Orang, Domba, Serigala, Wortol)) :-
    posisi(Orang),
    posisi(Domba),
    posisi(Serigala),
    posisi(Wortol), Orang == Serigala, Domba \= Wortol.

step((Orang1, Domba, Serigala, Wortol), 'Orang', (Orang2, Domba, Serigala, Wortol)) :-
    periksa((Orang2, Domba, Serigala, Wortol)),
    Orang1 \= Orang2.

step((Orang1, Domba1, Serigala, Wortol), 'Domba', (Orang2, Domba2, Serigala, Wortol)) :-
    periksa((Orang2, Domba2, Serigala, Wortol)),
    Orang1 \= Orang2, Domba1 \= Domba2.

step((Orang1, Domba, Serigala1, Wortol), 'Serigala', (Orang2, Domba, Serigala2, Wortol)) :-
    periksa((Orang2, Domba, Serigala2, Wortol)),
    Orang1 \= Orang2, Serigala1 \= Serigala2.

step((Orang1, Domba, Serigala, Wortol1), 'Wortol', (Orang2, Domba, Serigala, Wortol2)) :-
    periksa((Orang2, Domba, Serigala, Wortol2)),
    Orang1 \= Orang2, Wortol1 \= Wortol2.

reachable(S, _,[], S).
reachable(S, Visited, [Step|Steps], G) :-
    step(S, Step, Tmp), not(member(Tmp, Visited)),
    reachable(Tmp, [Tmp|Visited], Steps, G).

start((kiri,kiri,kiri,kiri)).
goal((kanan,kanan,kanan,kanan)).

solve(Steps) :- start(S), goal(G), reachable(S, [], Steps, G).
```

Line: 46

Hasil Trace

SWI-Prolog (AMD64, Multi-threaded, version 9.0.4)

File Edit Settings Run Debug Help

```
Welcome to SWI-Prolog (threaded, 64 bits, version 9.0.4)
SWI-Prolog comes with ABSOLUTELY NO WARRANTY. This is free software.
Please run ?- license. for legal details.

For online help and background, visit https://www.swi-prolog.org
For built-in help, use ?- help(Topic). or ?- apropos(Word).

?-
% d:/Download/kelompok3_level1.pl compiled 0.00 sec, 14 clauses
?- solve(Steps).
Steps = ['Domba', 'Orang', 'Serigala', 'Domba', 'Wortol', 'Orang', 'Domba'] ,

?- trace.
true.

[trace]  ?- step((kiri, kiri, kiri, kiri), 'Domba', (kanan, kanan, kiri, kiri)).
   Call: (10) step((kiri, kiri, kiri, kiri), 'Domba', (kanan, kanan, kiri, kiri)) ? creep
   Call: (11) periksa((kanan, kanan, kiri, kiri)) ? creep
   Call: (12) posisi(kanan) ? creep
   Exit: (12) posisi(kanan) ? creep
   Call: (12) posisi(kanan) ? creep
   Exit: (12) posisi(kanan) ? creep
   Call: (12) posisi(kiri) ? creep
   Exit: (12) posisi(kiri) ? creep
   Call: (12) posisi(kiri) ? creep
   Exit: (12) posisi(kiri) ? creep
   Call: (12) kanan==kanan ? creep
   Exit: (12) kanan==kanan ? creep
   Exit: (11) periksa((kanan, kanan, kiri, kiri)) ? creep
   Call: (11) kiri\=kanan ? creep
   Exit: (11) kiri\=kanan ? creep
   Call: (11) kiri\=kanan ? creep
   Exit: (11) kiri\=kanan ? creep
   Exit: (10) step((kiri, kiri, kiri, kiri), 'Domba', (kanan, kanan, kiri, kiri)) ? creep
true .

[trace]  ?- step((kanan, kanan, kiri, kiri), 'Orang', (kiri, kanan, kiri, kiri)).
   Call: (10) step((kanan, kanan, kiri, kiri), 'Orang', (kiri, kanan, kiri, kiri)) ? creep
   Call: (11) periksa((kiri, kanan, kiri, kiri)) ? creep
   Call: (12) posisi(kiri) ? creep
   Exit: (12) posisi(kiri) ? creep
   Call: (12) posisi(kanan) ? creep
   Exit: (12) posisi(kanan) ? creep
   Call: (12) posisi(kiri) ? creep
   Exit: (12) posisi(kiri) ? creep
   Call: (12) posisi(kiri) ? creep
   Exit: (12) posisi(kiri) ? creep
   Call: (12) kiri==kanan ? creep
   Fail: (12) kiri==kanan ? creep
   Redo: (11) periksa((kiri, kanan, kiri, kiri)) ? creep
   Call: (12) posisi(kiri) ? creep
   Exit: (12) posisi(kiri) ? creep
   Call: (12) posisi(kanan) ? creep
   Exit: (12) posisi(kanan) ? creep
   Call: (12) posisi(kiri) ? creep
   Exit: (12) posisi(kiri) ? creep
   Call: (12) posisi(kiri) ? creep
   Exit: (12) posisi(kiri) ? creep
   Call: (12) kiri==kiri ? creep
   Exit: (12) kiri==kiri ? creep
   Call: (12) kanan\=kiri ? creep
   Exit: (12) kanan\=kiri ? creep
   Exit: (11) periksa((kiri, kanan, kiri, kiri)) ? creep
   Call: (11) kanan\=kiri ? creep
   Exit: (11) kanan\=kiri ? creep
   Exit: (10) step((kanan, kanan, kiri, kiri), 'Orang', (kiri, kanan, kiri, kiri)) ? creep
true .
```

```
[trace]  ?- step((kiri, kanan, kiri, kiri), 'Orang', (kanan, kanan, kiri, kiri)).
   Call: (10) step((kiri, kanan, kiri, kiri), 'Orang', (kanan, kanan, kiri, kiri)) ? creep
   Call: (11) periksa((kanan, kanan, kiri, kiri)) ? creep
   Call: (12) posisi(kanan) ? creep
   Exit: (12) posisi(kanan) ? creep
   Call: (12) posisi(kanan) ? creep
   Exit: (12) posisi(kanan) ? creep
   Call: (12) posisi(kiri) ? creep
   Exit: (12) posisi(kiri) ? creep
   Call: (12) posisi(kiri) ? creep
   Exit: (12) posisi(kiri) ? creep
   Call: (12) kanan==kanan ? creep
   Exit: (12) kanan==kanan ? creep
   Exit: (11) periksa((kanan, kanan, kiri, kiri)) ? creep
   Call: (11) kiri\=kanan ? creep
   Exit: (11) kiri\=kanan ? creep
   Exit: (10) step((kiri, kanan, kiri, kiri), 'Orang', (kanan, kanan, kiri, kiri)) ? creep
true .

[trace]  ?- step((kiri, kanan, kiri, kiri), 'Serigala', (kanan, kanan, kanan, kiri)).
   Call: (10) step((kiri, kanan, kiri, kiri), 'Serigala', (kanan, kanan, kanan, kiri)) ? creep
   Call: (11) periksa((kanan, kanan, kanan, kiri)) ? creep
   Call: (12) posisi(kanan) ? creep
   Exit: (12) posisi(kanan) ? creep
   Call: (12) posisi(kanan) ? creep
   Exit: (12) posisi(kanan) ? creep
   Call: (12) posisi(kanan) ? creep
   Exit: (12) posisi(kanan) ? creep
   Call: (12) posisi(kiri) ? creep
   Exit: (12) posisi(kiri) ? creep
   Call: (12) kanan==kanan ? creep
   Exit: (12) kanan==kanan ? creep
   Exit: (11) periksa((kanan, kanan, kanan, kiri)) ? creep
   Call: (11) kiri\=kanan ? creep
   Exit: (11) kiri\=kanan ? creep
   Call: (11) kiri\=kanan ? creep
   Exit: (11) kiri\=kanan ? creep
   Exit: (10) step((kiri, kanan, kiri, kiri), 'Serigala', (kanan, kanan, kanan, kiri)) ? creep
true .

[trace]  ?- step((kanan, kanan, kanan, kiri), 'Domba', (kiri, kiri, kanan, kiri)).
   Call: (10) step((kanan, kanan, kanan, kiri), 'Domba', (kiri, kiri, kanan, kiri)) ? creep
   Call: (11) periksa((kiri, kiri, kanan, kiri)) ? creep
   Call: (12) posisi(kiri) ? creep
   Exit: (12) posisi(kiri) ? creep
   Call: (12) posisi(kiri) ? creep
   Exit: (12) posisi(kiri) ? creep
   Call: (12) posisi(kanan) ? creep
   Exit: (12) posisi(kanan) ? creep
   Call: (12) posisi(kiri) ? creep
   Exit: (12) posisi(kiri) ? creep
   Call: (12) kiri==kiri ? creep
   Exit: (12) kiri==kiri ? creep
   Exit: (11) periksa((kiri, kiri, kanan, kiri)) ? creep
   Call: (11) kanan\=kiri ? creep
   Exit: (11) kanan\=kiri ? creep
   Call: (11) kanan\=kiri ? creep
   Exit: (11) kanan\=kiri ? creep
   Exit: (10) step((kanan, kanan, kanan, kiri), 'Domba', (kiri, kiri, kanan, kiri)) ? creep
true .

[trace]  ?- step((kiri, kiri, kanan, kiri), 'Domba', (kanan, kanan, kanan, kiri)).
   Call: (10) step((kiri, kiri, kanan, kiri), 'Domba', (kanan, kanan, kanan, kiri)) ? creep
   Call: (11) periksa((kanan, kanan, kanan, kiri)) ? creep
   Call: (12) posisi(kanan) ? creep
   Exit: (12) posisi(kanan) ? creep
   Call: (12) posisi(kanan) ? creep
   Exit: (12) posisi(kanan) ? creep
   Call: (12) posisi(kanan) ? creep
   Exit: (12) posisi(kanan) ? creep
   Call: (12) posisi(kiri) ? creep
   Exit: (12) posisi(kiri) ? creep
   Call: (12) kanan==kanan ? creep
   Exit: (12) kanan==kanan ? creep
   Exit: (11) periksa((kanan, kanan, kanan, kiri)) ? creep
   Call: (11) kiri\=kanan ? creep
   Exit: (11) kiri\=kanan ? creep
   Call: (11) kiri\=kanan ? creep
   Exit: (11) kiri\=kanan ? creep
   Exit: (10) step((kiri, kiri, kanan, kiri), 'Domba', (kanan, kanan, kanan, kiri)) ? creep
true .

[trace]  ?- step((kiri, kiri, kanan, kiri), 'Wortol', (kanan, kiri, kanan, kanan)).
   Call: (10) step((kiri, kiri, kanan, kiri), 'Wortol', (kanan, kiri, kanan, kanan)) ? creep
   Call: (11) periksa((kanan, kiri, kanan, kanan)) ? creep
   Call: (12) posisi(kanan) ? creep
   Exit: (12) posisi(kanan) ? creep
   Call: (12) posisi(kiri) ? creep
   Exit: (12) posisi(kiri) ? creep
   Call: (12) posisi(kanan) ? creep
   Exit: (12) posisi(kanan) ? creep
   Call: (12) posisi(kanan) ? creep
   Exit: (12) posisi(kanan) ? creep
   Call: (12) kanan==kiri ? creep
   Fail: (12) kanan==kiri ? creep
   Redo: (11) periksa((kanan, kiri, kanan, kanan)) ? creep
   Call: (12) posisi(kanan) ? creep
   Exit: (12) posisi(kanan) ? creep
   Call: (12) posisi(kiri) ? creep
   Exit: (12) posisi(kiri) ? creep
   Call: (12) posisi(kanan) ? creep
   Exit: (12) posisi(kanan) ? creep
   Call: (12) posisi(kanan) ? creep
   Exit: (12) posisi(kanan) ? creep
   Call: (12) kanan==kanan ? creep
   Exit: (12) kanan==kanan ? creep
   Call: (12) kiri\=kanan ? creep
   Exit: (12) kiri\=kanan ? creep
   Exit: (11) periksa((kanan, kiri, kanan, kanan)) ? creep
   Call: (11) kiri\=kanan ? creep
   Exit: (11) kiri\=kanan ? creep
   Call: (11) kiri\=kanan ? creep
   Exit: (11) kiri\=kanan ? creep
   Exit: (10) step((kiri, kiri, kanan, kiri), 'Wortol', (kanan, kiri, kanan, kanan)) ? creep
true .

[trace]  ?- step((kanan,kiri,kanan,kanan),'Orang',(kiri,kiri,kanan,kanan)).
   Call: (10) step((kanan, kiri, kanan, kanan), 'Orang', (kiri, kiri, kanan, kanan)) ? creep
   Call: (11) periksa((kiri, kiri, kanan, kanan)) ? creep
   Call: (12) posisi(kiri) ? creep
   Exit: (12) posisi(kiri) ? creep
   Call: (12) posisi(kiri) ? creep
   Exit: (12) posisi(kiri) ? creep
   Call: (12) posisi(kanan) ? creep
   Exit: (12) posisi(kanan) ? creep
   Call: (12) posisi(kanan) ? creep
   Exit: (12) posisi(kanan) ? creep
   Call: (12) kiri==kiri ? creep
   Exit: (12) kiri==kiri ? creep
   Exit: (11) periksa((kiri, kiri, kanan, kanan)) ? creep
   Call: (11) kanan\=kiri ? creep
   Exit: (11) kanan\=kiri ? creep
   Exit: (10) step((kanan, kiri, kanan, kanan), 'Orang', (kiri, kiri, kanan, kanan)) ? creep
true .

[trace]  ?- step((kiri, kiri, kanan, kanan), 'Orang', (kanan, kiri, kanan, kanan)).
   Call: (10) step((kiri, kiri, kanan, kanan), 'Orang', (kanan, kiri, kanan, kanan)) ? creep
   Call: (11) periksa((kanan, kiri, kanan, kanan)) ? creep
   Call: (12) posisi(kanan) ? creep
   Exit: (12) posisi(kanan) ? creep
   Call: (12) posisi(kiri) ? creep
   Exit: (12) posisi(kiri) ? creep
   Call: (12) posisi(kanan) ? creep
   Exit: (12) posisi(kanan) ? creep
   Call: (12) posisi(kanan) ? creep
   Exit: (12) posisi(kanan) ? creep
   Call: (12) kanan==kiri ? creep
   Fail: (12) kanan==kiri ? creep
   Redo: (11) periksa((kanan, kiri, kanan, kanan)) ? creep
   Call: (12) posisi(kanan) ? creep
   Exit: (12) posisi(kanan) ? creep
   Call: (12) posisi(kiri) ? creep
   Exit: (12) posisi(kiri) ? creep
   Call: (12) posisi(kanan) ? creep
   Exit: (12) posisi(kanan) ? creep
   Call: (12) posisi(kanan) ? creep
   Exit: (12) posisi(kanan) ? creep
   Call: (12) kanan==kanan ? creep
   Exit: (12) kanan==kanan ? creep
   Call: (12) kiri\=kanan ? creep
   Exit: (12) kiri\=kanan ? creep
   Exit: (11) periksa((kanan, kiri, kanan, kanan)) ? creep
   Call: (11) kiri\=kanan ? creep
   Exit: (11) kiri\=kanan ? creep
   Exit: (10) step((kiri, kiri, kanan, kanan), 'Orang', (kanan, kiri, kanan, kanan)) ? creep
true

[trace]  ?- step((kiri, kiri, kanan, kanan), 'Domba', (kanan, kanan, kanan, kanan)).
   Call: (10) step((kiri, kiri, kanan, kanan), 'Domba', (kanan, kanan, kanan, kanan)) ? creep
   Call: (11) periksa((kanan, kanan, kanan, kanan)) ? creep
   Call: (12) posisi(kanan) ? creep
   Exit: (12) posisi(kanan) ? creep
   Call: (12) posisi(kanan) ? creep
   Exit: (12) posisi(kanan) ? creep
   Call: (12) posisi(kanan) ? creep
   Exit: (12) posisi(kanan) ? creep
   Call: (12) posisi(kanan) ? creep
   Exit: (12) posisi(kanan) ? creep
   Call: (12) kanan==kanan ? creep
   Exit: (12) kanan==kanan ? creep
   Exit: (11) periksa((kanan, kanan, kanan, kanan)) ? creep
   Call: (11) kiri\=kanan ? creep
   Exit: (11) kiri\=kanan ? creep
   Call: (11) kiri\=kanan ? creep
   Exit: (11) kiri\=kanan ? creep
   Exit: (10) step((kiri, kiri, kanan, kanan), 'Domba', (kanan, kanan, kanan, kanan)) ? creep
true .
```