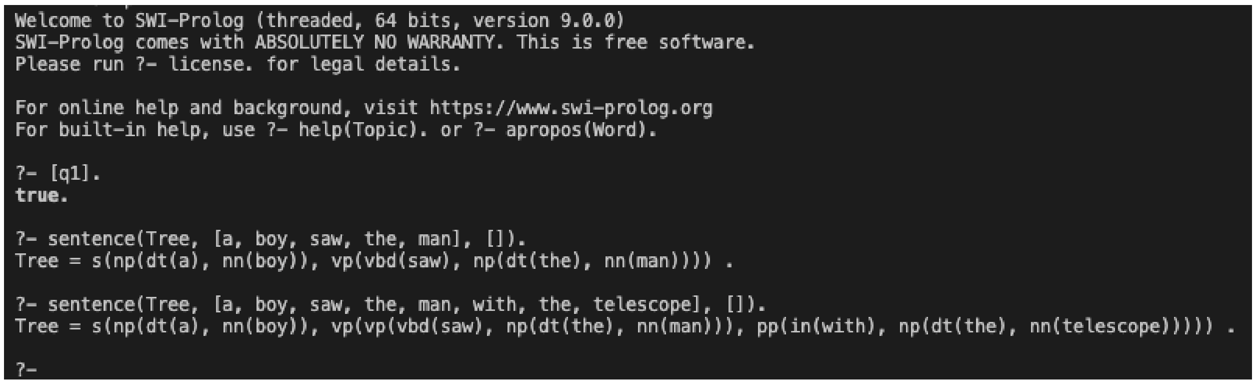
HW12

Q1: Transform g1.prolog into g1tree.prolog so it can parse.

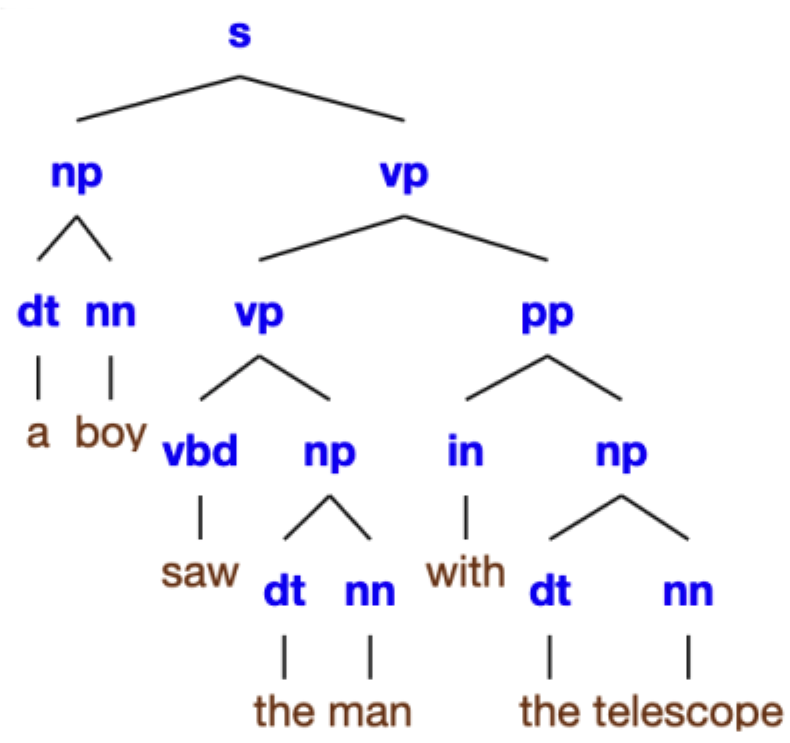
```
sentence(s(NP,VP))-->noun_part(NP),verb_part(VP).
noun_part(np(D,N))-->art(D),noun(N).
verb_part(vp(V,NP))-->verb(V),noun_part(NP).
verb_part(vp(VP,PP))-->verb_part(VP),prep_phase(PP).
prep_phase(pp(IN,NP))-->prep(IN),noun_part(NP).
prep(in(with))-->[with].
art(dt(the))-->[the].
art(dt(a))-->[a].
noun(nn(boy))-->[boy].
noun(nn(man))-->[man].
noun(nn(telescope))-->[telescope].
verb(vbd(saw))-->[saw].
```

Output:



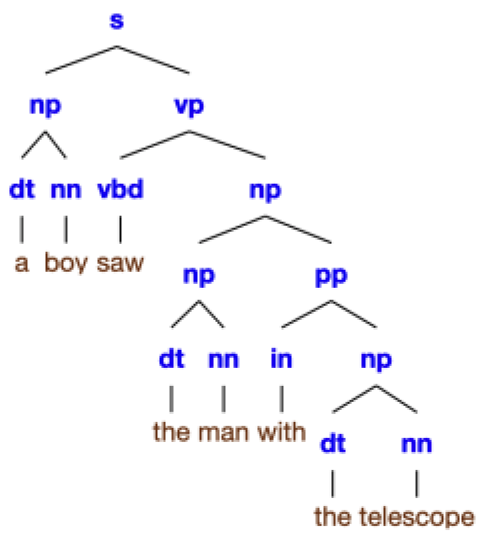
```
Welcome to SWI-Prolog (threaded, 64 bits, version 9.0.0)
SWI-Prolog comes with ABSOLUTELY NO WARRANTY. This is free software.
Please run ?- license. for legal details.

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

?- [q1].
true.

?- sentence(Tree, [a, boy, saw, the, man], []).
Tree = s(np(dt(a), nn(boy)), vp(vbd(saw), np(dt(the), nn(man)))) .

?- sentence(Tree, [a, boy, saw, the, man, with, the, telescope], []).
Tree = s(np(dt(a), nn(boy)), vp(vp(vbd(saw), np(dt(the), nn(man))), pp(in(with), np(dt(the), nn(telescope))))) .

?-
```

The tree generated by the online tool by the given above rules.



Q2: What new grammar rule does this parse add to g1.prolog?

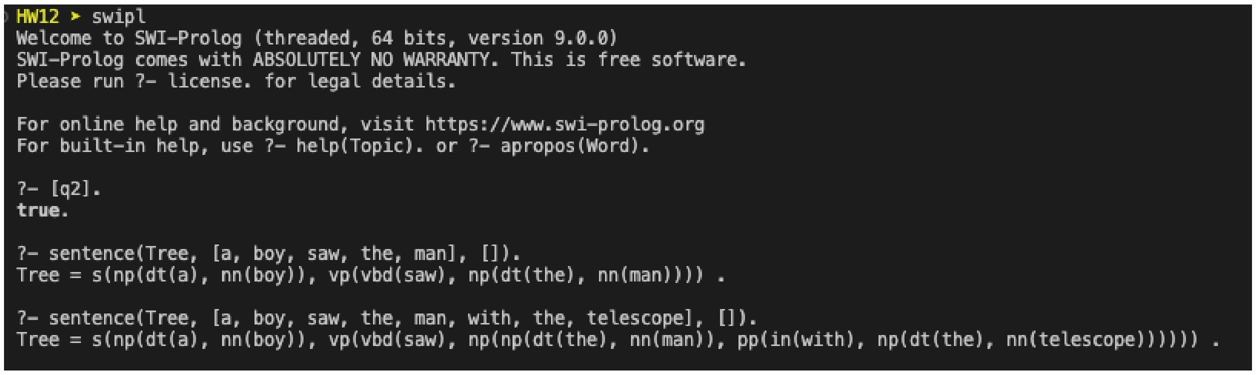
I will add NP -> NP, PP as a new to g1.prolog. The below tree was generated after that:



Here is my corresponding code after adding that rule:

```
sentence(s(NP,VP))-->noun_part(NP),verb_part(VP).
noun_part(np(D,N))-->art(D),noun(N).
noun_part(np(NP,PP))-->noun_part(NP),prep_part(PP).
verb_part(vp(V,NP))-->verb(V),noun_part(NP).
verb_part(vp(VP,PP))-->verb_part(VP),prep_part(PP).
prep_part(pp(IN,NP))-->prep(IN),noun_part(NP).
prep(in(with))-->[with].
art(dt(the))-->[the].
art(dt(a))-->[a].
noun(nn(boy))-->[boy].
noun(nn(man))-->[man].
noun(nn(telescope))-->[telescope].
verb(vbd(saw))-->[saw].
```

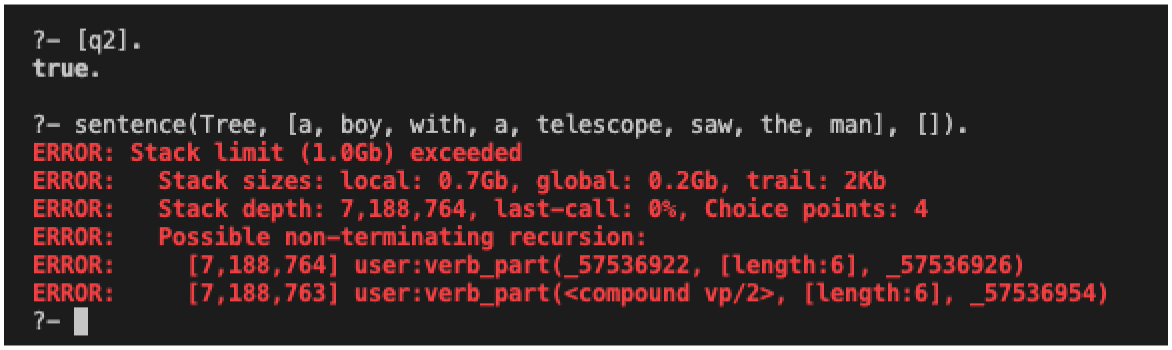
Output:



```
HW12 ➤ swipl
Welcome to SWI-Prolog (threaded, 64 bits, version 9.0.0)
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).

?- [q2].
true.

?- sentence(Tree, [a, boy, saw, the, man], []).
Tree = s(np(dt(a), nn(boy)), vp(vbd(saw), np(dt(the), nn(man)))) .

?- sentence(Tree, [a, boy, saw, the, man, with, the, telescope], []).
Tree = s(np(dt(a), nn(boy)), vp(vbd(saw), np(np(dt(the), nn(man)), pp(in(with), np(dt(the), nn(telescope)))))) .
```

Q3: What happens when you try: ?- s(Tree, [a, boy, with, a, telescope, saw, the, man], []).

The above query failed after I tried it, according to the Q2 code. Contrary to what was discussed in class, I think the explanation is that there aren't any restrictions for the array A that starts with a saw. It keeps broadening the NP rule but is unable to reach the PP.



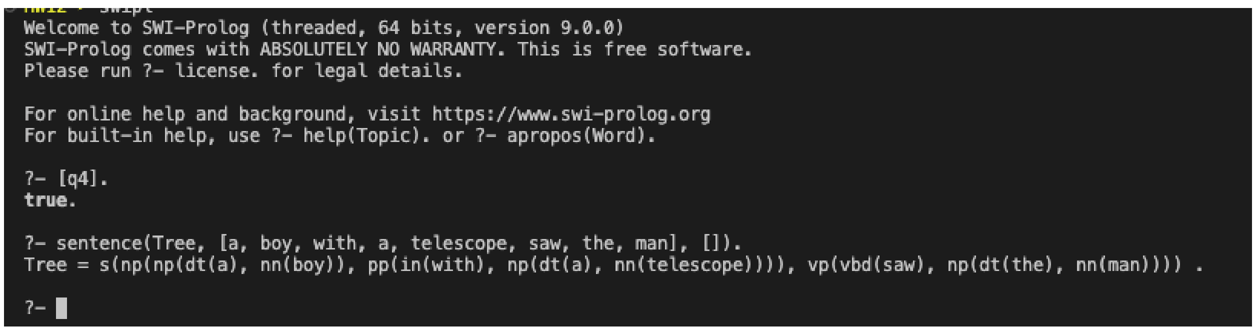
```
?- [q2].
true.

?- sentence(Tree, [a, boy, with, a, telescope, saw, the, man], []).
ERROR: Stack limit (1.0Gb) exceeded
ERROR:   Stack sizes: local: 0.7Gb, global: 0.2Gb, trail: 2Kb
ERROR:   Stack depth: 7,188,764, last-call: 0%, Choice points: 4
ERROR:   Possible non-terminating recursion:
ERROR:     [7,188,764] user:verb_part(_57536922, [length:6], _57536926)
ERROR:     [7,188,763] user:verb_part(<compound vp/2>, [length:6], _57536954)
?- 
```

Q4: Implement the left corner idea for VP.

After adding the corner case, my code is working fine and there is no infinite loop now.

```
sentence(s(NP,VP)) --> noun_part(NP) , verb_part(VP).
noun_part(np(D,N)) --> det(D), noun(N).
noun_part(np(NP,PP)) --> noun_part(NP), prep_part(PP).
verb_part(vp(V,NP)) --> verb(V), noun_part(NP).
verb_part(vp(VP,PP),A,B):- A = [saw|_], verb_part(VP, A, C), prep_part(PP, C, B).
prep_part(pp(IN, NP)) --> prep(IN), noun_part(NP).
prep(in(with)) --> [with].
det(dt(the)) --> [the].
det(dt(a)) --> [a].
noun(nn(boy)) --> [boy].
noun(nn(man)) --> [man].
noun(nn(telescope)) --> [telescope].
verb(vbd(saw)) --> [saw].
```

Output:



```
Welcome to SWI-Prolog (threaded, 64 bits, version 9.0.0)
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).

?- [q4].
true.

?- sentence(Tree, [a, boy, with, a, telescope, saw, the, man], []).
Tree = s(np(np(dt(a), nn(boy)), pp(in(with), np(dt(a), nn(telescope)))), vp(vbd(saw), np(dt(the), nn(man)))) .

?- 
```