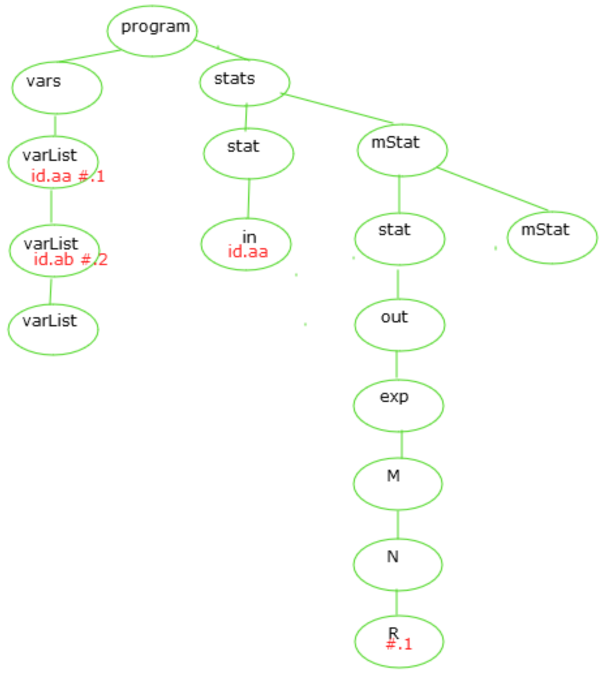
```
let aa = 1
ab = 2 .
main
  scan aa .
  print 1 .
end
```



Note:
You can store all tokens but only those needed to be stored, as explained elsewhere (ids, #s, and artithmetical/relational operators), are stored as they are processed.

If you left-factorized something like expression or VarList you would have extra nodes in the tree but the shape shoudl be the same. The shape, and the needed tokes, are what is important. Here they are not left-factorized but instead implemented with the trick as explained in suggestions.

On empty transitions you may have empty nodes or skip the nodes - here they are shown.

P2 output (preorder, - is one identation)

```
program
-vars
--varList id aa 1 #tk  1 1 // token, instance, line 1 if you process lines
---varList id ab 2 #tk 2  2
----varList
-stats
--stat
---stat
----in id aa 4
--mStat
---stat
----stat
-----out
------exp
-------M
--------N
---------R #tk 1 5
---mStat
```