

F -> with Ada.Text_IO; use Ada.Text_IO;
procedure *<ident>* is **D***

D -> type *<ident>*;
type *<ident>* is access *<ident>*
type *<ident>* is record **C**+ end record;
<ident>⁺, : *<type>* (:= **E**?);
procedure *<ident>* *<params>*? is **D***
begin **I** + end *<ident>*? ;
function *<ident>* **P** ? return *<type>* is **D***

C -> *<ident>*⁺, : **T**;

T -> *<ident>*
access *<ident>*

P -> (**P**⁺ ;)

P -> *<ident>*⁺, : **M**? **T**

M -> in | in out

E -> <entier> | <caractère> | true | false | null

(E)

A

E O E

not E | - E

new <ident>

<ident> (E⁺,)

character ' val (E)

I -> **A** := E

<ident> ;

<ident> (E⁺,) ;

return E? ;

begin I⁺ end ;

if E then I⁺ (elsif E then I⁺)* (else I⁺)? end if ;

for <ident> in reverse? E ... E loop I⁺ end loop ;

while E loop I⁺ end loop ;

O -> = | /= | < | <= | > | >= | + | - | * | /

rem | and | and then | or | or else

A -> <ident> | E.<ident>