Ejemplo

```
F:power(){
   float base;
   float n;
   float i, p;
   p = 1;
   for(i = 1; i <= n; ++i){
      p = p * base;
   }
   return p;
}
...
```

⇒

```
...
$F:power(){
   float base;
   float n;
   float i, p;
   p = 1;
   for(i = 1; i <= n; ++i){
      p = p * base;
   }
   return p;
}
```

$\langle retvp \rangle \rightarrow lit \mid V_n'$

$\langle retv \rangle \rightarrow \langle sign \rangle \langle retvp \rangle$

`return;` ➡ $*^{:ret}$

`return` $\langle retv_1 \rangle$`;` ➡

```
&R:ret = ⟨retv1⟩;
return;
```

Ejemplo

```
return 13;
```

⇒

```
$R.ret = 13;
return;
```

⇒ ... ⇒

```
...
*:ret
```

←:⟨*id*⟩

Señales que pone el expansor.

$$W \rightarrow V \mid \mathbf{N}$$

```
$W:nombre(){ ➡
    texto1
}
```

```
:def_subp:{
    #definec $R W
    texto1
    Z:i = *Y:top;          ←:ret
    :rest_vals:
    *                      ←:i
    1:np                   ←:npos
}
```

Ejemplo

```
$F:power(){
    float base;
    float n;
    float i, p;
    p = 1;
    for(i = 1; i <= n; ++i){
        p = p * base;
    }
    return p;
}
```

⇒

```
:def_subp:{
    #definec $R F
    float base;
    ...
        p = p * base;
    }
    return p;
    Z:67 = *Y:top;
    :rest_vals:            ←:ret
    *
    1:np                   ←:67
}                          ←:npos
```

:rest_vals:

:rest_vals: ➡

$Z_{:top}$--;
$Z_{.r1}$ = *$Y_{:top}$;
$Z_{:top}$--;
$Z_{.r2}$ = *$Y_{:top}$;
$Z_{:top}$--;
$Z_{.and}$ = *$Y_{:top}$;
$Z_{:top}$--;
$Z_{.or}$ = *$Y_{:top}$;
$Z_{:top}$--;
$Z_{.ob1}$ = *$Y_{:top}$;
$Z_{:top}$--;
$Z_{.op1}$ = *$Y_{:top}$;
$Z_{:top}$--;
$Z_{.brk}$ = *$Y_{:top}$;
$Z_{:top}$--;
$Z_{.sw}$ = *$Y_{:top}$;
$Z_{:top}$--;
$Z_{:top}$ = *$Y_{:top}$;

:def_subp:{ *texto*$_1$ }

Sea *m* el ⟨*natural*⟩ asociado a **:num_position:**. El expansor:

- asocia 1 a **:num_position:**.
- expande las macroinstrucciones en *texto*$_1$ hasta obtener ⟨*preinstrucciones*$_1$⟩.
- sustituye ←**:return** por una señal, sea ←***:k***, y todas las apariciones de **:return** en ⟨*preinstrucciones*$_1$⟩ por ***:k***.
- sustituye la última ⟨*preinstrucción*⟩, $1^{:np}$, por 1^n, donde *n* es el ⟨*natural*⟩ asociado a **:num_position:**.
- sustituye ←**:npos** por una señal, sea ←***:p***, y todas las apariciones de **:npos** en ⟨*preinstrucciones*$_1$⟩ por ***:p***.
- asocia *m* a **:num_position:**.

Expansiones finales 1”

Si se usan macroinstrucciones **:def_subp:**, el expansor, en las expansiones finales:

- antes de añadir la ⟨*preinstrucción*⟩ $1^{\text{:top}}$ añade una ⟨*preinstrucción*⟩ 1^{n}, donde *n* es el ⟨*natural*⟩ asociado a **:num_position:**.
- añade una señal ←**:$npos**. apuntando a esta ⟨*preinstrucción*⟩.
- sustituye las apariciones de **:npos** en cualquier ⟨*preinstrucción*⟩ por **:$npos**.
- por cada señal ←**:⟨*id*$_1$⟩** presente:
  - la sustituye por una señal ←**:⟨*natural*⟩**; sea ésta ←**:*i***.
  - sustituye las apariciones de **:⟨*id*$_1$⟩** en cualquier ⟨*preinstrucción*⟩ por **:*i***.

Llamada a subprogramas

$\langle argp\_y\rangle \rightarrow \langle natural\rangle \mid \mathbf{0} \mid \mathbf{Y}'_n \mid \&V_n\langle indf\rangle$

$\langle arg\_y\rangle \rightarrow \langle argp\_y\rangle \mid +\langle argp\_y\rangle$

$\langle argp\_f\rangle \rightarrow \langle racional\rangle \mid \mathbf{F}'_n$

$\langle arg\_f\rangle \rightarrow \langle signo\rangle\langle argp\_f\rangle$

$\langle arg\rangle \rightarrow \langle arg\_y\rangle \mid \langle arg\_f\rangle$

$\langle rest\_args\rangle \rightarrow \varepsilon \mid \mathbf{,}\langle arg\rangle\langle rest\_args\rangle$

$\langle args\rangle \rightarrow \varepsilon \mid \langle arg\rangle\langle rest\_args\rangle$