1. ⟨id⟩ ::= a | b | c | ... | z
⟨dig⟩ ::= 0 | 1 | 2 | ... | 9
⟨expr⟩ ::= () | ⟨dig⟩ | ⟨id⟩
| let ⟨id⟩ = ⟨expr⟩ in ⟨expr⟩
| ⟨expr⟩ ; ⟨expr⟩
| begin ⟨expr⟩ end

use example: let x = 5 in begin x ; x end

There are two different derivations

1) ⟨expr⟩
⇒ let ⟨id⟩ = ⟨expr⟩ in ⟨expr⟩
let x = ⟨expr⟩ in ⟨expr⟩
let x = ⟨dig⟩ in ⟨expr⟩
let x = 5 in ⟨expr⟩
let x = 5 in begin ⟨expr⟩ end
let x = 5 in begin x ; ⟨expr⟩ end
let x = 5 in begin x ; x end

2) ⟨expr⟩
let ⟨id⟩ = ⟨expr⟩ in ⟨expr⟩
let x = ⟨expr⟩ in ⟨expr⟩
let x = ⟨dig⟩ in ⟨expr⟩
let x = 5 in ⟨expr⟩
let x = 5 in ⟨dig⟩
let x = 5 in x

There are two different ways to derive the same strings, displaying ambiguity.

2. ⟨id⟩ ::= a | b | c | ... | z
⟨dig⟩ ::= 0 | 1 | 2 | ... | 9
⟨expr⟩ ::= ⟨term⟩ | ⟨let⟩
⟨term⟩ ::= () | ⟨dig⟩ | ⟨id⟩ | ⟨semi⟩ | ⟨begin⟩
⟨let⟩ ::= let ⟨id⟩ = ⟨expr⟩ in ⟨term⟩
⟨semi⟩ ::= ⟨term⟩ ; ⟨term⟩
⟨begin⟩ ::= begin ⟨expr⟩ end

3. <expr>

<let>

let <id> = <exp> in <term>

a b <seml>

<term> ; <term>

<id> <id>

a c