program
    definice_funkce definice_funkcí **;** **EOF**

definice_funkcí
    definice_funkce definice_funkcí
    nic

definice_funkce
    function identifier (formální_parametry **)** sekvence_deklarací sekvence_příkazů **end**

formální_paramtery
    identifier formální_parametry_z
    nic

formální_parametry_z
    , identifier formální_parametry_z
    nic

sekvence_deklarací
    local identifier deklarace_local_z sekvence_deklarací
    nic

deklarace_local_z
    = literál **;**
    ;

literál
    number
    string
    nil
    true
    false

sekvence_příkazů
    příkaz **;** sekvence_příkazů
    nic

příkaz
    identifier = assign_z
    write (seznam_výrazů **)**
    if výraz **then** sekvence_příkazů **else** sekvence_příkazů **end**
    while výraz **do** sekvence_příkazů **end**
    return výraz

assign_z
    read (literál **)**
    výraz
    identifier (seznam_literálů_a_proměnných **)**

seznam_literálů_a_proměnných
	literál seznam_literálů_a_proměnných_a
	idenifier seznam_literálů_a_proměnných_a
	nic

seznam_literálů_a_proměnných_a
	, seznam_literálů_a_proměnných_z seznam_literálů_a_proměnných_a
	nic

seznam_literálů_a_proměnných_z
	literál
	identifier

***** Tohle popisuje gramatiku s ε-pravidly (značeno nic)
***** Mám důkaz, že je to **LL** (pokud jsem ho neudělal blbě)
***** ALE! Neřeší to výrazy, které neumím udělat, vznikla by mi tam kolize s identifikátory kvůli pravidlům:
assign_z
	výraz
	identifier (seznam_literálů_a_proměnných **)**
Kde ve výrazu může být identifikátor, nicméně rozdíl je v tom, že se jedná o ID z různých tabulek, takže by se to teoreticky dalo rozlišit takto.
Další možné řešení je, že by volání funkce bylo součástí výrazu. To je tuším nějaké rozšíření.
Tohle se bude řešit asi až po přednášce na výrazy.

// na výrazy kašlem zatím, to je tu jen z nouze, dále je jen bordel:
seznam_výrazů
    výraz seznam_výrazu_z
    nic

seznam_výrazů_z
    , výraz seznam_výrazu_z
    nic

výraz
    literál
    identifier
    (výraz **)**
    výraz operátor výraz
    huh??
    Výrazy jsou tvořeny literály, již definovanými proměnnými, závorkami nebo výrazy tvořenými binárními aritmetickými a relačními operátory. Na rozdíl od jazyka Lua nejsou implicitní konverze povoleny až na specifikované výjimky

operátor
    plus
    minus
    div
    mul
    power
    strconcat
    less
    great
    less_eq
    great_eq
    equal
    not_equal