```
chunk ::= {stat [`;´]} [laststat [`;´]]

block ::= chunk

stat ::=  varlist `=´ explist |
          functioncall |
          do block end |
          while exp do block end |
          repeat block until exp |
          if exp then block {elseif exp then block} [else block] end |
          for Name `=´ exp `,´ exp [`,´ exp] do block end |
          for namelist in explist do block end |
          function funcname funcbody |
          local function Name funcbody |
          local namelist [`=´ explist]

laststat ::= return [explist] | break

funcname ::= Name {`.´ Name} [`:´ Name]

varlist ::= var {`,´ var}

namelist ::= Name {`,´ Name}

explist ::= {exp `,´} exp

args ::=  `(´ [explist] `)´ | tableconstructor | String

function ::= function funcbody

funcbody ::= `(´ [parlist] `)´ block end

parlist ::= namelist [`,´ `...´] | `...´

tableconstructor ::= `{´ [fieldlist] `}´

fieldlist ::= field {fieldsep field} [fieldsep]

field ::= `[´ exp `]´ `=´ exp | Name `=´ exp | exp

fieldsep ::= `,´ | `;´
```

```
binop ::= `+´ | `-´ | `*´ | `/´ | `^´ | `%´ | `..´ |
          `<´ | `<=´ | `>´ | `>=´ | `==´ | `~=´ |
          and | or

unop ::= `-´ | not | `#´

var ::== Name var2
        '(' exp ')' prefixexp2   '[' exp ']' var2
        '(' exp ')' prefixexp2   '.' Name var2

var2::== prefixexp2   '[' exp ']'  var2
        prefixexp2      '.' Name          var2
        e

prefixexp ::== var prefixexp2
        '(' exp ')' prefixexp2

prefixexp2 ::== args prefixexp2
                `:´ Name args prefixexp2
                e

functioncall ::=  prefixexp args | prefixexp `:´ Name args


exp ::=  nil exp2 | false exp2 | true exp2 | Number exp2 | String exp2 | `...´ exp2
| function exp2 |  prefixexp exp2 | tableconstructor exp2 | unop exp exp2

exp2 ::== binop exp exp2
        e
```