Laboratorio di Linguaggi Formali e Traduttori
LFT lab T4, a.a. 2022/2023

Docente: Luigi Di Caro

Analisi sintattica

(parte seconda)

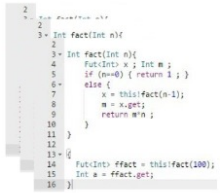
Laboratorio di Linguaggi Formali e Traduttori
LFT lab T4, a.a. 2022/2023

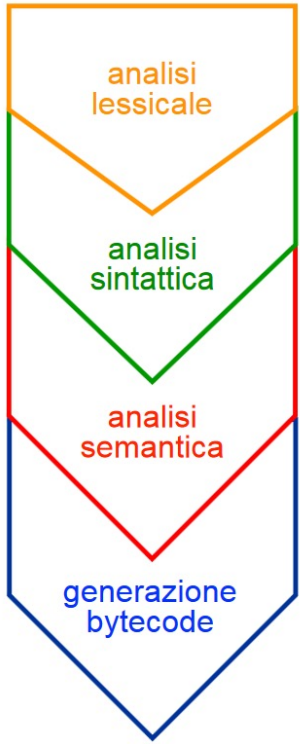
Docente: Luigi Di Caro

25 Novembre – Giornata internazionale per l'eliminazione della violenza sulle donne

- https://www.unito.it/ateneo/organizzazione/organi-di-ateneo/comitato-unico-di-garanzia/progetti-e-attivita/giornata
- https://www.youtube.com/watch?v=foGkjzWNG4A
- https://www.un.org/en/observances/ending-violence-against-women-day

Analisi sintattica





identifica le parole della descrizione

identifica come le parole sono legate l'una all'altra

controlla che il programma non violi delle richieste di consistenza

genera codice per la Java Virtual Machine

output

Il **bytecode** è interpretato dalla JVM

Analisi sintattica

- Obiettivo dell'esercizio 3.1: scrivere un programma per l'analisi sintattica per **espressioni aritmetiche** semplici, scritte in notazione infissa.

$$
\begin{array}{rcl}
\langle start \rangle & ::= & \langle expr \rangle \ \texttt{EOF} \\
\\
\langle expr \rangle & ::= & \langle term \rangle \ \langle exprp \rangle \\
\\
\langle exprp \rangle & ::= & + \ \langle term \rangle \ \langle exprp \rangle \\
 & | & - \ \langle term \rangle \ \langle exprp \rangle \\
 & | & \varepsilon \\
\\
\langle term \rangle & ::= & \langle fact \rangle \ \langle termp \rangle \\
\\
\langle termp \rangle & ::= & \star \ \langle fact \rangle \ \langle termp \rangle \\
 & | & / \ \langle fact \rangle \ \langle termp \rangle \\
 & | & \varepsilon \\
\\
\langle fact \rangle & ::= & (\ \langle expr \rangle\ )\ |\ \texttt{NUM}
\end{array}
$$

Analisi sintattica

- Obiettivo dell'esercizio 3.1: scrivere un programma per l'analisi sintattica per espressioni aritmetiche semplici, scritte in notazione infissa.

- **Esercizio 3.2: nuova grammatica!**

$$
\begin{array}{rcl}
\langle prog \rangle & ::= & \langle statlist \rangle \ \texttt{EOF} \\
\langle statlist \rangle & ::= & \langle stat \rangle \ \langle statlistp \rangle \\
\langle statlistp \rangle & ::= & \texttt{;} \ \langle stat \rangle \ \langle statlistp \rangle \ \mid \ \varepsilon \\
\langle stat \rangle & ::= & \texttt{assign} \ \langle expr \rangle \ \texttt{to} \ \langle idlist \rangle \\
 & \mid & \texttt{print} \ \texttt{[} \ \langle exprlist \rangle \ \texttt{]} \\
 & \mid & \texttt{read} \ \texttt{[} \ \langle idlist \rangle \ \texttt{]} \\
 & \mid & \texttt{while} \ \texttt{(} \ \langle bexpr \rangle \ \texttt{)} \ \langle stat \rangle \\
 & \mid & \texttt{conditional} \ \texttt{[} \ \langle optlist \rangle \ \texttt{]} \ \texttt{end} \\
 & \mid & \texttt{conditional} \ \texttt{[} \ \langle optlist \rangle \ \texttt{]} \ \texttt{else} \ \langle stat \rangle \ \texttt{end} \\
 & \mid & \texttt{\{} \ \langle statlist \rangle \ \texttt{\}} \\
\langle idlist \rangle & ::= & \texttt{ID} \ \langle idlistp \rangle \\
\langle idlistp \rangle & ::= & \texttt{,} \ \texttt{ID} \ \langle idlistp \rangle \ \mid \ \varepsilon \\
\langle optlist \rangle & ::= & \langle optitem \rangle \ \langle optlistp \rangle \\
\langle optlistp \rangle & ::= & \langle optitem \rangle \ \langle optlistp \rangle \ \mid \ \varepsilon \\
\langle optitem \rangle & ::= & \texttt{option} \ \texttt{(} \ \langle bexpr \rangle \ \texttt{)} \ \texttt{do} \ \langle stat \rangle \\
\langle bexpr \rangle & ::= & \texttt{RELOP} \ \langle expr \rangle \ \langle expr \rangle \\
\langle expr \rangle & ::= & \texttt{+} \ \texttt{(} \ \langle exprlist \rangle \ \texttt{)} \ \mid \ \texttt{-} \ \langle expr \rangle \ \langle expr \rangle \\
 & \mid & \texttt{*} \ \texttt{(} \ \langle exprlist \rangle \ \texttt{)} \ \mid \ \texttt{/} \ \langle expr \rangle \ \langle expr \rangle \\
 & \mid & \texttt{NUM} \ \mid \ \texttt{ID} \\
\langle exprlist \rangle & ::= & \langle expr \rangle \ \langle exprlistp \rangle \\
\langle exprlistp \rangle & ::= & \texttt{,} \ \langle expr \rangle \ \langle exprlistp \rangle \ \mid \ \varepsilon
\end{array}
$$

Analisi sintattica

- **prog** rappresenta un programma
- **EOF** è l'end of file
- **statlist** è una sequenza di comandi
  - stat e statlistp
    - **stat** indica un comando
    - **statlistp** è un elenco di comandi potenzialmente vuoto

$$
\begin{array}{rcl}
\langle prog \rangle & ::= & \langle statlist \rangle \ \texttt{EOF} \\
\langle statlist \rangle & ::= & \langle stat \rangle \ \langle statlistp \rangle \\
\langle statlistp \rangle & ::= & \texttt{;} \ \langle stat \rangle \ \langle statlistp \rangle \ \mid \ \varepsilon \\
\langle stat \rangle & ::= & \texttt{assign} \ \langle expr \rangle \ \texttt{to} \ \langle idlist \rangle \\
 & \mid & \texttt{print} \ \texttt{[} \ \langle exprlist \rangle \ \texttt{]} \\
 & \mid & \texttt{read} \ \texttt{[} \ \langle idlist \rangle \ \texttt{]} \\
 & \mid & \texttt{while} \ \texttt{(} \ \langle bexpr \rangle \ \texttt{)} \ \langle stat \rangle \\
 & \mid & \texttt{conditional} \ \texttt{[} \ \langle optlist \rangle \ \texttt{]} \ \texttt{end} \\
 & \mid & \texttt{conditional} \ \texttt{[} \ \langle optlist \rangle \ \texttt{]} \ \texttt{else} \ \langle stat \rangle \ \texttt{end} \\
 & \mid & \texttt{\{} \ \langle statlist \rangle \ \texttt{\}} \\
\langle idlist \rangle & ::= & \texttt{ID} \ \langle idlistp \rangle \\
\langle idlistp \rangle & ::= & \texttt{,} \ \texttt{ID} \ \langle idlistp \rangle \ \mid \ \varepsilon \\
\langle optlist \rangle & ::= & \langle optitem \rangle \ \langle optlistp \rangle \\
\langle optlistp \rangle & ::= & \langle optitem \rangle \ \langle optlistp \rangle \ \mid \ \varepsilon \\
\langle optitem \rangle & ::= & \texttt{option} \ \texttt{(} \ \langle bexpr \rangle \ \texttt{)} \ \texttt{do} \ \langle stat \rangle \\
\langle bexpr \rangle & ::= & \texttt{RELOP} \ \langle expr \rangle \ \langle expr \rangle \\
\langle expr \rangle & ::= & \texttt{+} \ \texttt{(} \ \langle exprlist \rangle \ \texttt{)} \ \mid \ \texttt{-} \ \langle expr \rangle \ \langle expr \rangle \\
 & \mid & \texttt{*} \ \texttt{(} \ \langle exprlist \rangle \ \texttt{)} \ \mid \ \texttt{/} \ \langle expr \rangle \ \langle expr \rangle \\
 & \mid & \texttt{NUM} \ \mid \ \texttt{ID} \\
\langle exprlist \rangle & ::= & \langle expr \rangle \ \langle exprlistp \rangle \\
\langle exprlistp \rangle & ::= & \texttt{,} \ \langle expr \rangle \ \langle exprlistp \rangle \ \mid \ \varepsilon
\end{array}
$$

Analisi sintattica

- **statlistp** è un elenco di comandi potenzialmente vuoto
  - ; è un separatore di comandi. Serve se c'è un ulteriore comando
  - fine dell'elenco dei comandi → **epsilon**
- **stat** è un singolo comando
  - diverse produzioni per diversi tipi di comandi

$$
\begin{array}{rcl}
\langle prog \rangle & ::= & \langle statlist \rangle \ \texttt{EOF} \\
\langle statlist \rangle & ::= & \langle stat \rangle \ \langle statlistp \rangle \\
\langle statlistp \rangle & ::= & \texttt{;} \ \langle stat \rangle \ \langle statlistp \rangle \ \mid \ \varepsilon \\
\langle stat \rangle & ::= & \texttt{assign} \ \langle expr \rangle \ \texttt{to} \ \langle idlist \rangle \\
 & \mid & \texttt{print [} \ \langle exprlist \rangle \ \texttt{]} \\
 & \mid & \texttt{read [} \ \langle idlist \rangle \ \texttt{]} \\
 & \mid & \texttt{while (} \ \langle bexpr \rangle \ \texttt{)} \ \langle stat \rangle \\
 & \mid & \texttt{conditional [} \ \langle optlist \rangle \ \texttt{]} \ \texttt{end} \\
 & \mid & \texttt{conditional [} \ \langle optlist \rangle \ \texttt{]} \ \texttt{else} \ \langle stat \rangle \ \texttt{end} \\
 & \mid & \texttt{\{} \ \langle statlist \rangle \ \texttt{\}} \\
\langle idlist \rangle & ::= & \texttt{ID} \ \langle idlistp \rangle \\
\langle idlistp \rangle & ::= & \texttt{,} \ \texttt{ID} \ \langle idlistp \rangle \ \mid \ \varepsilon \\
\langle optlist \rangle & ::= & \langle optitem \rangle \ \langle optlistp \rangle \\
\langle optlistp \rangle & ::= & \langle optitem \rangle \ \langle optlistp \rangle \ \mid \ \varepsilon \\
\langle optitem \rangle & ::= & \texttt{option (} \ \langle bexpr \rangle \ \texttt{)} \ \texttt{do} \ \langle stat \rangle \\
\langle bexpr \rangle & ::= & \texttt{RELOP} \ \langle expr \rangle \ \langle expr \rangle \\
\langle expr \rangle & ::= & \texttt{+ (} \ \langle exprlist \rangle \ \texttt{)} \ \mid \ \texttt{-} \ \langle expr \rangle \ \langle expr \rangle \\
 & \mid & \texttt{* (} \ \langle exprlist \rangle \ \texttt{)} \ \mid \ \texttt{/} \ \langle expr \rangle \ \langle expr \rangle \\
 & \mid & \texttt{NUM} \ \mid \ \texttt{ID} \\
\langle exprlist \rangle & ::= & \langle expr \rangle \ \langle exprlistp \rangle \\
\langle exprlistp \rangle & ::= & \texttt{,} \ \langle expr \rangle \ \langle exprlistp \rangle \ \mid \ \varepsilon
\end{array}
$$

Analisi sintattica

- **stat** è un singolo comando
  - **assign** serve per l'assegnamento diretto. Assegniamo ad **<idlist>** il valore dell'espressione **expr**
  - **print** stampa un elenco di espressioni
  - **read** - legge in input valori e assegna
  - **while** - classico ciclo while con espressione booleana
  - **conditional / optlist / optitem** - condizionale in stile switch di java

$$
\begin{array}{rcl}
\langle prog \rangle & ::= & \langle statlist \rangle \ \texttt{EOF} \\
\langle statlist \rangle & ::= & \langle stat \rangle \ \langle statlistp \rangle \\
\langle statlistp \rangle & ::= & \texttt{;} \ \langle stat \rangle \ \langle statlistp \rangle \ \mid \ \varepsilon \\
\langle stat \rangle & ::= & \texttt{assign} \ \langle expr \rangle \ \texttt{to} \ \langle idlist \rangle \\
 & \mid & \texttt{print} \ \texttt{[} \ \langle exprlist \rangle \ \texttt{]} \\
 & \mid & \texttt{read} \ \texttt{[} \ \langle idlist \rangle \ \texttt{]} \\
 & \mid & \texttt{while} \ \texttt{(} \ \langle bexpr \rangle \ \texttt{)} \ \langle stat \rangle \\
 & \mid & \texttt{conditional} \ \texttt{[} \ \langle optlist \rangle \ \texttt{]} \ \texttt{end} \\
 & \mid & \texttt{conditional} \ \texttt{[} \ \langle optlist \rangle \ \texttt{]} \ \texttt{else} \ \langle stat \rangle \ \texttt{end} \\
 & \mid & \texttt{\{} \ \langle statlist \rangle \ \texttt{\}} \\
\langle idlist \rangle & ::= & \texttt{ID} \ \langle idlistp \rangle \\
\langle idlistp \rangle & ::= & \texttt{,} \ \texttt{ID} \ \langle idlistp \rangle \ \mid \ \varepsilon \\
\langle optlist \rangle & ::= & \langle optitem \rangle \ \langle optlistp \rangle \\
\langle optlistp \rangle & ::= & \langle optitem \rangle \ \langle optlistp \rangle \ \mid \ \varepsilon \\
\langle optitem \rangle & ::= & \texttt{option} \ \texttt{(} \ \langle bexpr \rangle \ \texttt{)} \ \texttt{do} \ \langle stat \rangle \\
\langle bexpr \rangle & ::= & \texttt{RELOP} \ \langle expr \rangle \ \langle expr \rangle \\
\langle expr \rangle & ::= & \texttt{+} \ \texttt{(} \ \langle exprlist \rangle \ \texttt{)} \ \mid \ \texttt{-} \ \langle expr \rangle \ \langle expr \rangle \\
 & \mid & \texttt{*} \ \texttt{(} \ \langle exprlist \rangle \ \texttt{)} \ \mid \ \texttt{/} \ \langle expr \rangle \ \langle expr \rangle \\
 & \mid & \texttt{NUM} \ \mid \ \texttt{ID} \\
\langle exprlist \rangle & ::= & \langle expr \rangle \ \langle exprlistp \rangle \\
\langle exprlistp \rangle & ::= & \texttt{,} \ \langle expr \rangle \ \langle exprlistp \rangle \ \mid \ \varepsilon
\end{array}
$$

Analisi sintattica

- **bexpr** è un'espressione booleana che coinvolge un **RELOP** e <u>due</u> **expr**
  - RELOP → {==, <>, <= , >= , < , >}
    - esempio: > *x 0*
- **expr** ha più produzioni
  - **NUM** costante numerica
  - **ID** identificatore
  - oppure coinvolge operatori aritmetici
    - semplici: sottrazioni e divisioni
    - più complesse: moltiplicazioni e somme

$$
\begin{array}{rcl}
\langle prog \rangle & ::= & \langle statlist \rangle \ \texttt{EOF} \\
\langle statlist \rangle & ::= & \langle stat \rangle \ \langle statlistp \rangle \\
\langle statlistp \rangle & ::= & \texttt{;} \ \langle stat \rangle \ \langle statlistp \rangle \ \mid \ \varepsilon \\
\langle stat \rangle & ::= & \texttt{assign} \ \langle expr \rangle \ \texttt{to} \ \langle idlist \rangle \\
 & \mid & \texttt{print [} \ \langle exprlist \rangle \ \texttt{]} \\
 & \mid & \texttt{read [} \ \langle idlist \rangle \ \texttt{]} \\
 & \mid & \texttt{while (} \ \langle bexpr \rangle \ \texttt{)} \ \langle stat \rangle \\
 & \mid & \texttt{conditional [} \ \langle optlist \rangle \ \texttt{] end} \\
 & \mid & \texttt{conditional [} \ \langle optlist \rangle \ \texttt{] else} \ \langle stat \rangle \ \texttt{end} \\
 & \mid & \texttt{\{} \ \langle statlist \rangle \ \texttt{\}} \\
\langle idlist \rangle & ::= & \texttt{ID} \ \langle idlistp \rangle \\
\langle idlistp \rangle & ::= & \texttt{,} \ \texttt{ID} \ \langle idlistp \rangle \ \mid \ \varepsilon \\
\langle optlist \rangle & ::= & \langle optitem \rangle \ \langle optlistp \rangle \\
\langle optlistp \rangle & ::= & \langle optitem \rangle \ \langle optlistp \rangle \ \mid \ \varepsilon \\
\langle optitem \rangle & ::= & \texttt{option (} \ \langle bexpr \rangle \ \texttt{)} \ \texttt{do} \ \langle stat \rangle \\
\langle bexpr \rangle & ::= & \texttt{RELOP} \ \langle expr \rangle \ \langle expr \rangle \\
\langle expr \rangle & ::= & \texttt{+ (} \ \langle exprlist \rangle \ \texttt{)} \ \mid \ \texttt{-} \ \langle expr \rangle \ \langle expr \rangle \\
 & \mid & \texttt{* (} \ \langle exprlist \rangle \ \texttt{)} \ \mid \ \texttt{/} \ \langle expr \rangle \ \langle expr \rangle \\
 & \mid & \texttt{NUM} \ \mid \ \texttt{ID} \\
\langle exprlist \rangle & ::= & \langle expr \rangle \ \langle exprlistp \rangle \\
\langle exprlistp \rangle & ::= & \texttt{,} \ \langle expr \rangle \ \langle exprlistp \rangle \ \mid \ \varepsilon
\end{array}
$$

Analisi sintattica

- **exprlist** e **exprlistp** è il solito elenco di elementi
  - in questo caso gli elementi sono espressioni **expr**

$$
\begin{array}{rcl}
\langle prog \rangle & ::= & \langle statlist \rangle \ \texttt{EOF} \\
\langle statlist \rangle & ::= & \langle stat \rangle \ \langle statlistp \rangle \\
\langle statlistp \rangle & ::= & \texttt{;} \ \langle stat \rangle \ \langle statlistp \rangle \ \mid \ \varepsilon \\
\langle stat \rangle & ::= & \texttt{assign} \ \langle expr \rangle \ \texttt{to} \ \langle idlist \rangle \\
 & \mid & \texttt{print} \ \texttt{[} \ \langle exprlist \rangle \ \texttt{]} \\
 & \mid & \texttt{read} \ \texttt{[} \ \langle idlist \rangle \ \texttt{]} \\
 & \mid & \texttt{while} \ \texttt{(} \ \langle bexpr \rangle \ \texttt{)} \ \langle stat \rangle \\
 & \mid & \texttt{conditional} \ \texttt{[} \ \langle optlist \rangle \ \texttt{]} \ \texttt{end} \\
 & \mid & \texttt{conditional} \ \texttt{[} \ \langle optlist \rangle \ \texttt{]} \ \texttt{else} \ \langle stat \rangle \ \texttt{end} \\
 & \mid & \texttt{\{} \ \langle statlist \rangle \ \texttt{\}} \\
\langle idlist \rangle & ::= & \texttt{ID} \ \langle idlistp \rangle \\
\langle idlistp \rangle & ::= & \texttt{,} \ \texttt{ID} \ \langle idlistp \rangle \ \mid \ \varepsilon \\
\langle optlist \rangle & ::= & \langle optitem \rangle \ \langle optlistp \rangle \\
\langle optlistp \rangle & ::= & \langle optitem \rangle \ \langle optlistp \rangle \ \mid \ \varepsilon \\
\langle optitem \rangle & ::= & \texttt{option} \ \texttt{(} \ \langle bexpr \rangle \ \texttt{)} \ \texttt{do} \ \langle stat \rangle \\
\langle bexpr \rangle & ::= & \texttt{RELOP} \ \langle expr \rangle \ \langle expr \rangle \\
\langle expr \rangle & ::= & \texttt{+} \ \texttt{(} \ \langle exprlist \rangle \ \texttt{)} \ \mid \ \texttt{-} \ \langle expr \rangle \ \langle expr \rangle \\
 & \mid & \texttt{*} \ \texttt{(} \ \langle exprlist \rangle \ \texttt{)} \ \mid \ \texttt{/} \ \langle expr \rangle \ \langle expr \rangle \\
 & \mid & \texttt{NUM} \ \mid \ \texttt{ID} \\
\langle exprlist \rangle & ::= & \langle expr \rangle \ \langle exprlistp \rangle \\
\langle exprlistp \rangle & ::= & \texttt{,} \ \langle expr \rangle \ \langle exprlistp \rangle \ \mid \ \varepsilon
\end{array}
$$

Analisi sintattica

Esercizio 3.2, cominciamo con un esempio

```
read(x);
print(x)
```

Analisi sintattica

Esercizio 3.2, cominciamo con un esempio

```
read[x];
print[x]
```

- ; non sempre si deve inserire a fine di ogni riga. E' un separatore di due comandi
- tra parentesi, la read ha un identificatore
  - nota: non abbiamo bisogno di dichiarazioni nel nostro linguaggio

Analisi sintattica

read[x]

- guardiamo la read
  - Legge valori in input che vengono assegnati ad identificatori (in questo caso, solo uno: x)

Analisi sintattica

print[x]

- guardiamo la print
  - Stampa il valore di una serie di espressioni (in questo caso, solo il valore di un identificatore: x)

Analisi sintattica

- guardiamo la assign
  - Assegna il valore di un’espressione ad una lista di identificatori (in questo caso, solo ad x)

```
read[x];
print[x];
assign 0 to x
```

Analisi sintattica

- guardiamo il while
  - Statement per ciclo

```
read[x];
print[x];
assign 0 to x;
while (> x 0) { ... }
```

Analisi sintattica

- **Pronti per scrivere il codice del parser 3.2**

$$
\begin{array}{rcl}
\langle prog \rangle & ::= & \langle statlist \rangle \ \texttt{EOF} \\
\langle statlist \rangle & ::= & \langle stat \rangle \ \langle statlistp \rangle \\
\langle statlistp \rangle & ::= & \texttt{;} \ \langle stat \rangle \ \langle statlistp \rangle \ \mid \ \varepsilon \\
\langle stat \rangle & ::= & \texttt{assign} \ \langle expr \rangle \ \texttt{to} \ \langle idlist \rangle \\
 & \mid & \texttt{print} \ [\ \langle exprlist \rangle \ ] \\
 & \mid & \texttt{read} \ [\ \langle idlist \rangle \ ] \\
 & \mid & \texttt{while} \ (\ \langle bexpr \rangle \ ) \ \langle stat \rangle \\
 & \mid & \texttt{conditional} \ [\ \langle optlist \rangle \ ] \ \texttt{end} \\
 & \mid & \texttt{conditional} \ [\ \langle optlist \rangle \ ] \ \texttt{else} \ \langle stat \rangle \ \texttt{end} \\
 & \mid & \{ \ \langle statlist \rangle \ \} \\
\langle idlist \rangle & ::= & \texttt{ID} \ \langle idlistp \rangle \\
\langle idlistp \rangle & ::= & \texttt{,} \ \texttt{ID} \ \langle idlistp \rangle \ \mid \ \varepsilon \\
\langle optlist \rangle & ::= & \langle optitem \rangle \ \langle optlistp \rangle \\
\langle optlistp \rangle & ::= & \langle optitem \rangle \ \langle optlistp \rangle \ \mid \ \varepsilon \\
\langle optitem \rangle & ::= & \texttt{option} \ (\ \langle bexpr \rangle \ ) \ \texttt{do} \ \langle stat \rangle \\
\langle bexpr \rangle & ::= & \texttt{RELOP} \ \langle expr \rangle \ \langle expr \rangle \\
\langle expr \rangle & ::= & \texttt{+} \ (\ \langle exprlist \rangle \ ) \ \mid \ \texttt{-} \ \langle expr \rangle \ \langle expr \rangle \\
 & \mid & \texttt{*} \ (\ \langle exprlist \rangle \ ) \ \mid \ \texttt{/} \ \langle expr \rangle \ \langle expr \rangle \\
 & \mid & \texttt{NUM} \ \mid \ \texttt{ID} \\
\langle exprlist \rangle & ::= & \langle expr \rangle \ \langle exprlistp \rangle \\
\langle exprlistp \rangle & ::= & \texttt{,} \ \langle expr \rangle \ \langle exprlistp \rangle \ \mid \ \varepsilon
\end{array}
$$