Osservazioni Esercizio 15 Novembre

Paola Giannini

Commenti generali

- Pattern
  - Sia per gli identificatori che per gli interi e i float alcuni hanno dato soluzioni molto più generali del richiesto. Va bene, ma non è cosi semplice!
  - Quando chiedevo la classe lessicale rappresentata volevo che sceglieste fra una delle seguenti
    - Identificatori, Costanti /Letterali, Parole chiave, Delimitatori, Operatori

    Forse la richiesta non era chiara. Alcuni hanno interpretato questo nel senso dello specifico token.
  - Qualcuno ha confuso i token `TYINT` con `INT` e `TYFLOAT` con `FLOAT`.
- Implementazione tokens tutti avete proposto di avere una classe con un tipo enumerato, ma ci sarebbero alternative!

Alcune classificazioni scorrette

Token	Pattern	errore
TYPEINT	int	Costante intera. NO è una Parola Chiave
TYINT	int$	Tipo intero. NO è una Parola Chiave (e il $?)
ASSIGN	=	Istruzione di assegnamento. NO è un Operatore
PRINT	print	Istruzione di stampa. NO è una Parola Chiave
PRINT	print	Operatore. NO è una Parola Chiave
PRINT	print$	Stampa a schermo. NO è una Parola Chiave (e il $?)
SEMI	;	Separatore di Riga. NO è solo Separatore

Alcuni pattern scorretti

Token	Pattern	errore
ID	tempa,tempb	Identificatori. Questo NON è il pattern
INT	[0-9]	genera una sola cifra
INT	[0-9]+.[0-9]	genera una sola cifra decimale
IntNum	[1-9] | ([0-9])* | 0	genera interi che iniziano con 0
FloatNum	((+ | -)([0-9]*),([0-9])*)	il separatore è '.' e non devono iniziare per 0
FLOAT	(([0-9])+(.([0-9])+f?)?)	anche una costante intera matcha questo pattern
FLOAT	(([0-9])+(.([0-9])+f?)?)	0 sarebbe un FLOAT
Float	3,2	Costante float. Questo NON è il pattern
Print	print id	Il pattern è SOLO 'print'

Alcuni pattern più generali

Token	Pattern	errore
ID	[a-zA-Z]([a-zA-Z]|[0-9]|_)*	OK ma un id può anche iniziare con _
ID	(_ | [A-z])([A-z]|[0-9]|_)*	Molto ingegnoso!!!!!!
FloatNum	((+ | -)([0-9]*),([0-9])*)	I float possono avere il segno
INT	-?(0 | [1-9][0-9]*)	Perchè solo il segno -

Pattern MOLTO complicati

ID	(([a-z]|[A-Z])+(_|[0-9]|[a-z]|[A-Z])*)(?<!print|int|float)	IDENTIFICATORE
COMM	((?s)(\/[*]((?![*]\/).)*[*]\/))|(\/\/((?!(\n|\r)).|\t)*)	COMMENTI

Problema con il segno prima dei numeri

- Non si può riconoscere a livello lessicale (con un automa a stati finiti) il fatto che + o - siano parte del letterale oppure di una espressione binaria!

Come è risolto questo nel linguaggi di programmazione?

- + e - sono OPERATORI UNARI! 
- Nell'analisi lessicale + e - generano sempre un token operatore.
- Quando avviene il riconoscimento del fatto che siano unari o binari?
- A livello di ANALISI SINTATTICA cioè specificato da una grammatica Context Free!

Una soluzione corretta

Token	Pattern	Classe rappresentata
INT	[1-9][0-9]* | 0	Costante
FLOAT	([1-9][0-9]* | 0).[0-9]{1,5}	Costante
ID	[a-z]+	Identificatore
TYINT	int	Parola chiave
TYFLOAT	float	Parola chiave
ASSIGN	=	Operatore
PRINT	print	Operatore
PLUS	+	Operatore
MINUS	-	Operatore
SEMI	;	Delimitatore
EOF	(char) -1	Fine Input

CARATTERI DA IGNORARE: ' ', '\n', '\t', '\r'

Implementazione alternativa: Una classe per ogni tipo di Token

```
public abstract class Token {......} // contiene i campi
                                     // comuni a tutti i token
public class TkFloat {......}//non ha campi aggiuntivi

public class TkId {private String val;...}//contiene la stringa
                                          //corrispondente a Id
...............
```