# Corso: Fondamenti, Linguaggi e Traduttori

Paola Giannini

Esercizio: analizzatore lessicale del linguaggio ac



### Input del Compilatore

#### Programma Sorgente ac

```
int tempa;
tempa = 5;
float tempb = tempa + 3.2;
tempb = tempb * 7;
print tempb;
```

dichiarazione variabile intero assegnamento di costante intera a variabile dichiarazione e inizializzazione variabile float assegnamento di espressione prodotto a variabile stampa variabile



#### Esercizio 1

Identificate i token del linguaggio e per ogni token scrivete l'espressione regolare che ne è il pattern. Ricordate che sappiamo che i token corrisponderanno a

- parole chiave
- identificatori
- costanti
- operatori e delimitatori
- eof

Elencare anche i caratteri da ignorare?



### Cosa dovete fare

• Riempite la seguente tabella con, il nome del Token, l'espressione regolare che descrive il token e quale classe del linguaggio rappresenta.

Token	Pattern	Classe rappresentata
INT	??????	costante intera
ID	?????	identitificatore
MINUS	????	operatore
SEMI	???	separatore
E0F	(char) -1	Fine Input

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



## Sintassi per espressioni regolari

Usate la sintassi estesa (quella di Java) delle espressioni regolari, cioè:

- Un carattere a denota la stringa che contiene quel carattere, cioè "a"
- Una sequenza significa concatenazione ab significa a seguito b

Alcuni caratteri, operatori, o parentesi hanno un significato speciale. I principali sono:

- | : alternanza a | b significa a oppure b
- \*: ripetizione a\* significa zero o più a ("", a, aa,...)
- + : ripetizione a+ significa una o più a (a, aa,...)
- ? : facoltatività a? significa "" oppure a
- ( ): ragruppamento (per dare scope ad un operatore; esempio (ab)\* significa ab ripetuti zero o piu' volte
- [ ]: denota un insieme, ad esempio [abc] identifica uno fra a, b o c sarebbe
   (a | b | c)
- - un range, ad esempio [0-9] : range da 0 a 9 (una cifra)
- expReg{n,m} la ripetizione almeno n volte e meno di m+1 del pattern expReg (cioè da n a m estremi inclusi)



# Esercizio 2: Rappresentazione dei Token

I token possono contenere anche informazioni aggiuntive. Ad esempio

- la riga del codice nella quale appare la stringa corrispondente; questo è utile, perchè, i token sono l'unico input del Parser. Se vogliamo segnalare la riga di un errore ne abbiamo bisogno!
- la colonna. Meno importante perchè le righe sono corte!

Come possono essere rappresentati i Token in Java?

