

2. Analisi lessicale

Implementazione in Java di un lexer
per un semplice linguaggio di
programmazione

Esercizio 2.2

- Consideriamo la seguente nuova definizione di identificatori): un identificatore è composto da una sequenza non vuota di **lettere**, **numeri**, ed il **simbolo di 'underscore'** **_** che:
 - **non comincia** con un **numero** e che
 - **non** può essere composto **solo dal simbolo** **_**. Più precisamente, gli identificatori corrispondono all'espressione regolare:

$$\left([a - zA - Z] \mid (-(-)^*[a - zA - Z0 - 9]) \right) \left([a - zA - Z0 - 9] \mid - \right)^*$$

- **Estendere il metodo lexical_scan** per gestire identificatori che corrispondono alla **nuova definizione**.

Esercizio 2.3

- Estendere il metodo `lexical_scan` in modo tale che possa trattare la **presenza di commenti** nel file di input. I commenti possono essere scritti in due modi:
 - commenti delimitati con `/*` e `*/` ;
 - commenti che iniziano con `//` e che terminano con un a capo oppure con EOF.
- I commenti devono essere **ignorati** dal programma per l'analisi lessicale;
- in altre parole, per le parti dell'input che contengono commenti, non deve essere generato nessun token.

Esercizio 2.3

- Ad esempio, consideriamo il seguente input.

```
/* calcolare la velocita' */  
d:=300; // distanza  
t:=10; // tempo  
print(d*t)
```

- L'output del programma per l'analisi lessicale sarà

```
<257, d >  
<262, := >  
<256, 300 >  
<59 >  
<257, t >  
<262, := >  
<256, 10 >  
<59 >  
<263, print >  
<40 >  
<257, d >  
<42 >  
<257, t >  
<41 >  
<-1 >
```

Esercizio 2.3

- Oltre alle coppie di simboli `/*`, `*/` e `//`, un commento può contenere simboli che non fanno parte del pattern di nessun token, ad esempio:
 - `/*@#?*/ o`
 - `/* calcolare la velocita' */` (apostrofo!)
- Se un commento di forma `/* ... */` è aperto ma non chiuso prima della fine del file deve essere segnalato un errore
 - ad esempio il caso di input

```
d:=300; /*distanza
```

Esercizio 2.3

- Si noti che ci possono essere **due commenti di seguito** non separati da nessun token, ad esempio:

```
d:=300; /*distanza*// *da Torino a Lione*/
```

Esercizio 2.3

- Inoltre la coppia di simboli `*/`, se scritta al di fuori di un commento, deve essere trattata dal lexer come il segno di moltiplicazione seguito dal segno di divisione
 - ad esempio, per l'input
$$x*/y$$
 - l'output sarà `<257, x > <42> <47> <257, y>`
- In altre parole, l'idea è che in questo caso la sequenza di simboli `*/` non verrà interpretata come un commento da saltare ma come una sequenza dei due token menzionati (moltiplicazione e divisione).

FAQ

Salve,

Sono una studentessa del suo laboratorio di Linguaggi Formali e Traduttori Corso A.

Vorrei chiederle delle delucidazioni riguardo all'esercizio 2.2 del Lexer.

Mi chiedevo se, nella definizione di identificatore, sono incluse anche le parole che contengono numeri al suo interno.

Per esempio, "abcd1234bs" viene riconosciuta come "acbd1234bs" o come "abcd" "1234" "bs"?

E invece, "1234abcd" come viene riconosciuta?

E invece "0245"?