Linguaggi e Traduttori

Esercitazione di laboratorio n° 2

1 Esercizio

Realizzare uno scanner per il linguaggio C. Lo scanner dovrà:

- Eliminare i commenti tipici del linguaggio C (/* e */)
- Riconoscere i simboli: “{“, “}”, “(“, “)”, “[“, “]”, “+”, “-”, “*”, “/”, “=”, “;”, “.”, “,”, “<”, “>”, “&”, “|”, “!”.
- Riconoscere alcune parole chiavi del C: `int`, `double`, `if`, `else`, `while`, `print`
- Riconoscere i numeri interi e double
- Eventuali `#include`, spazi, tabulazioni e newline dovranno essere scartati

Lo scanner dovrà fornire in output le unità riconosciute (i TOKEN) stampandoli separati da uno spazio. Per quanto riguarda gli interi, i numeri double e gli identificatori, dovrà stampare oltre al TOKEN riconosciuto anche il valore (Es. `INT:30`)

Il linguaggio C riconosciuto, come riportato nell’esempio, è una versione semplificata in cui non esistono le funzioni e le variabili possono essere dichiarate solo ad inizio file.

NB: Lo scanner realizzato per questa esercitazione verrà utilizzato nelle prossime esercitazioni per costruire un compilatore completo capace di compilare programmi simili a quello riportato nell’esempio che segue.

1.1 Esempi file di input

Si osservi l’esempio. Dato il seguente file di ingresso:

```
/* Esempio algoritmo di ordinamento Bubble sort */

double x[5];
int i, j;
double swap;
int pos;

/* Inizializzazione vettore */
x[0] = -2.0;
x[1] = -3.0;
x[2] = 3.0;
x[3] = 5.0;
x[4] = 2.5;

/* Bubble sort */
pos = 5;
while(pos > 0){
  i = 0;
  while (i < pos - 1){
    j = i + 1;
    if (x[i] > x[j]){
      swap = x[j];
      x[j] = x[i];
      x[i] = swap;
    }
    i = i + 1;
  }
  pos = pos-1;

}

/* Stampa risultati */
i = 0;
while(i<5){
  print (x[i]);
  i = i + 1;
}
```

1.2 Esempi di output

dovrà essere prodotto il seguente output:

```
DOUBLE_TYPE ID:x QO INT:5 QC PV INT_TYPE ID:i VIR ID:j
PV DOUBLE_TYPE ID:swap PV INT_TYPE ID:pos PV ID:x QO
INT:0 UG MEN DOBULE:2.0 PV ID:x QO INT:1 QC ...
...
ID:pos UG INT:5 PV WHILE TO ID:pos MAG INT:0 TC GO ID:i
UG INT:0 PV WHILE TO ID:i MIN ID:pos MEN ID:1 TC GO ID:j
...
```

2 Esercizio

Scrivere un analizzatore lessicale mediante JFLEX in grado di riconoscere gli elementi principali di un documento HTML. Un documento HTML è costituito da un testo ASCII annotato mediante opportune parole chiave.

Tutte le parole chiave sono racchiuse tra i simboli “<” e “>”. All’interno di tali simboli possono essere presenti anche eventuali modificatori e parametri delle parole chiave. Per le parole chiave ed i parametri è indifferente

l'uso di lettere maiuscole o minuscole. I due caratteri delimitatori con il loro contenuto costituiscono un tag. Le parole chiave sono costituite da caratteri alfanumerici, e possono iniziare con un carattere alfabetico. Inoltre, i tag di chiusura possono essere preceduti dal carattere "/". Un documento HTML può anche contenere dei commenti, che iniziano con la serie di caratteri "<!−−" e terminano con i caratteri "−− >". Tra le varie parole chiave che possono comparire, è richiesto di riconoscere esplicitamente le parole chiave "head", "body", "html", "title", "table", "h1", "h2", "h3", "h4".

L'analizzatore lessicale deve produrre in uscita il documento HTML in ingresso, depurato dei commenti. Inoltre, in coda deve stampare una serie di statistiche:

- il numero totale di tag che compaiono;
- il numero di tag table, h1, h2, h3 e h4 (e dei corrispondenti tag di chiusura).

Suggerimenti

- Utilizzare la direttiva `%caseless` di jflex per generare uno scanner case insensitive
- Utilizzare gli stati per distinguere commenti e tag dal testo generico

2.1 Esempio file di input

```
<HTML><HEAD><TITLE>Prova</TITLE></HEAD>
<BODY>
<!-- .... <table>Finta tabella (da non contare)</table> -->
<H1>Titolo_1</h1>
<h2>Titolo_1_1</H2>
Testo <b>vario</b>
<H1>Titolo_2</h1>
<h2>Titolo_2_1</H2>
<table border=2><tr><td>Idem</td></tr></table>
<a href="top.html"><img src="/img/top.gif" alt="top"></a>
<h2>Titolo_2_2</H2>
<table border=0><tr><td>
<table border=0><tr><td> Tabella annidata livello1</td></tr>
</table>
</table>
<!-- I tag che seguono indicano una lista non numerata -->
<ul>
<li><a href="pippo.htm">pippo</a>
<li><a href="pluto.htm">pluto<i>pi&ugrave;</i>pippo</a>
</ul>
<table border=0><tr><td>
<table border=0><tr><td>
<table border=0><tr><td>Tabel la annidata livello 2</td></tr>
</table>
</td></tr>
</table>
</table>
<img src="/img/bottom.gif " height="10" width="10" alt="bottom">
<hr>
```

2.2 Esempi di output

```
<HTML><HEAD><TITLE>Prova</TITLE></HEAD>
<BODY>
<H1>Titolo_1</h1>
<h2>Titolo_1_1</H2>
Testo <b>vario</b>
<H1>Titolo_2</h1>
<h2>Titolo_2_1</H2>
<table border=2><tr><td>Idem</td></tr></table>
<a href="top.html"><img src="/img/top.gif " alt="top"></a>
<h2>Titolo_2_2</H2>
<table border=0><tr><td>
<table border=0><tr><td>Tabella annidata livello1</td></tr>
</table>
</table>
```

```
<ul>
<li><a href="pippo.htm">pippo</a>
<li><a href="pluto.htm">pluto<i>pi&ugrave;</i>pippo</a>
</ul>
<table border=0><tr><td>
<table border=0><tr><td>
<table border=0><tr><td>Tabel la annidata livello2</td></tr>
</table>
</td></tr>
</table>
</table>
<img src="/img/bottom.gif " height="10" width="10" alt="bottom">
<hr>
</body></HTML>

Numero totale di tag: 67
Numero di tag table: 12
Numero di tag h1: 4
Numero di tag h2: 6
Numero di tag h3: 0
Numero di tag h4: 0
```