Linguaggi e Traduttori

Esercitazione di laboratorio n° 1

1 Esercizio

Date le seguenti definizioni regolari relative alla descrizione di un filename con eventuale path (secondo il formato DOS) scrivere, utilizzando il programma JFLEX, un scanner in grado di riconoscere un pathname corretto.

1.1 Definizioni Regolari

```
PathFileName → ( Drive : )? (\)? (PathName \ )* FileName (. FileType)?
Drive        → letter
PathName     → id
FileName     → id
FileType     → id
Id           → (letter | digit )( letter | digit )*
```

letter: tutte le lettere dell'alfabeto e i caratteri speciali tranne: slash, backslash, due punti, asterisco, punto interrogativo, doppie virgolette, minore, maggiore, simbolo OR.
digit: le cifre da 0 a 9

1.2 Esempi

```
A:\DIR1\DIR2\FILE1.EXE
E:\FILE1.BAT
DIR1\DIR2\FILE2
\UNIX\FILE.C
A:README.TXT
PIPPO
```

2 Esercizio

Scrivere le definizioni regolari ed il corrispondente programma JFLEX per riconoscere un sottoinsieme degli URL definiti da HTTP. Tale sottoinsieme deve comprendere:

- Il riconoscimento di più di uno schema (quindi http:, ftp:, gopher: e simili).
- I domini espressi come nomi (ad es. www.polito.it) o come indirizzi IP (ad es. 130.192.4.4).
- L'uso di porte differenti da quella di default per lo schema corrente (dominio seguito da ":" e dal numero della porta alternativa).
- L'uso di ancore all'interno di file HTML (nome del file seguito da "#" e dal nome dell'ancora).
- Sequenze di escape (cioè il carattere "%" seguito da una coppia di cifre esadecimali) in qualsiasi posizione.

2.1 Esempi di URL validi

Stringhe di caratteri riconosciute dall'analizzatore lessicale come FileName:

```
http://www.miosito.it/file.html
ftp://10.9.9.71/prova.zip
http://altro.sito.com/%7Eutente/index.html#rif33
nntp://news.sito.ch:8181/data/
```

3 Esercizio

Scrivere un analizzatore lessicale mediante un programma JFLEX che, dati in ingresso uno o più sorgenti C, riconosca tutti gli elementi propri di tale linguaggio (costanti numeriche, stringhe, direttive del prepocessore, keyword e commenti) e produca in uscita un file HTML in cui ciascun elemento risulti evidenziato con un colore diverso. Tutti i file in ingresso devono comparire in un unico file HTML preceduti dal nome del sorgente di origine. Si ricorda che in HTML i colori vengono espressi con un "#" seguito da tre coppie di cifre esadecimali, ciascuna delle quali rappresenta una componente fondamentale della terna RGB (quindi "#FF0000" è un rosso acceso, "#0000FF" è un blu e così via).

3.1 Esempio

Dato il seguente file di ingresso (di nome `main.c`):

```
#include <stdio.h>
int prova(int i) {
  if (i==1) return 1;
  /* questo è un commento */
  else return 0;
}
```

si ottiene in uscita quanto segue:

```
<HTML>
<BODY bgcolor="#FFFFFF">
<H2>main.c</H2>
<CODE>
<FONT COLOR="#00FF00">#include &lt;stdio.h&gt;</FONT>
<BR>
<FONT COLOR="#0000FF">int</FONT> prova(
<FONT COLOR="#0000FF">int</FONT> i)<BR>
{<BR>
<FONT COLOR="#0000FF">if</FONT>
(i==<FONT COLOR="#FF0000">1</FONT>)
<FONT COLOR="#0000FF">return</FONT>
<FONT COLOR="#FF0000">1</FONT>;<BR>
<FONT COLOR="#C0C0C0">/* questo è un commento */
</FONT><BR>
<FONT COLOR="#0000FF">else</FONT>
<FONT COLOR="#0000FF">return</FONT>
<FONT COLOR="#FF0000">0</FONT>;<BR>
}<BR>
</CODE></BODY></HTML>
```