```
/************************************* LEX ***************************************
** introduce in C uso di stringhe vincolate da espressioni regolari (E.R.) e permette **
** di controllare se e come una stringa soddisfa i patterns definiti per le E.R.      **
**      Dato un file di tipo .lex, LEX genera un file lex.yy.c contenente             **
**              una routine yylex (come main) ed                                      **
**              alcune tabelle di analisi (opportunamente codificate in C)            **
**      La routine yylex si comporta come un                                          **
**              driver di analisi lessicale                                           **
**              il cui lessico e' definito con il formalismo previsto per i files .lex**
*********************************************************************************/


/************************************* file .lex *********************************
** un file .lex contiene tre sezioni separate da %%:                                 **
**      {definitions}                                                                **
**      %%                                                                           **
**      {rules}                                                                      **
**      %%                                                                           **
**      {auxiliary procedures C}                                                     **
**                                                                                   **
** le sezioni hanno la struttura e funzionalita' seguenti                            **
**      definitions                                                                  **
**              <name> <E.R. su caratteri o names gia' definiti>                     **
**              introduce categorie lessicali e associata definizione                **
**      rules                                                                        **
**              <E.R.> <blocco di comandi C>                                         **
**              mette in corrispondenza                                              **
**                      la scansione di una stringa utilizzando una E.R.             **
**                      con l'esecuzione di un blocco C                              **
**                              la porzione di stringa che appartiene al linguaggio  **
**                              definito da E.R. e' associata alla variabile:        **
**                                      yytext                                       **
**      auxiliary                                                                    **
**              <procedura C>                                                        **
**              default e': main() {yylex();}                                        **
*********************************************************************************/


/****************** usare LEX come analizzatore lessicale: file .lex *****************
** Definiamo un file .lex contenente le sole prime due sezioni:                      **
**      {definitions}                                                                **
**      %%                                                                           **
**      {rules}                                                                      **
** nelle sezioni poniamo:                                                            **
**      definitions                                                                  **
**              <name> <E.R. su caratteri o names gia' definiti>                     **
**              le definizioni delle categorie lessicali                             **
**      rules                                                                        **
**              <E.R.> <blocco di comandi C>                                         **
**              la corrispondenza tra categoria lessicale e token generato           **
*********************************************************************************/



/***************************** COMPILARE ED ESEGUIRE (in UNIX) ***********************
** prodotto un file .lex, ad esempio prova.lex,                                      **
**      lex prova.lex                                                                **
** genera un file yy.lex.c di tipo c                                                 **
**      cc lex.yy.c -ll                                                              **
** produce un file a.out eseguibile                                                  **
** l'esecuzione di:                                                                  **
**      a.out                                                                        **
** cede il controllo al main che nel nostro caso e' la routine yylex()               **
**      la routine:                                                                  **
**              legge dall'input la sequenza di caratteri                            **
**              applica le regole                                                    **
**              produce l'output calcolato dal codice delle regole                   **
**                                                                                   **
** un esempio per prova.lex e' il seguente:                                          **
**      l'imput e'                                                                   **
**              pigreco   :=3.14   ; temp1:=pigreco*(temp1+2);                       **
**      l'output prodotto e'                                                         **
**      <IDE,pigreco> <:=,-><FRACT,3.14> <;,-> <IDE,temp1><:=,-><IDE,pigreco><OP2,*><(,- **
** ><IDE,temp1><OP2,+><NUM,2><),-><;,->                                              **
*********************************************************************************/



digit [0-9]     /* Ordinamento totale sui caratteri ASCCII */
digits [0-9]+   /* [0-9] e' la notazione LEX per la E.R. (0|1|2|3|4|5|6|7|8||9) */
ide [a-zA-Z][a-zA-Z0-9]*
separatore [ \t\n]*
rational {digits}"."{digits}
bin [+*/]

%%
{digits} {printf("<NUM,%s>", yytext);}
{rational} {printf("<FRACT,%s>", yytext);}
":="|";"|while|"("|")"|"{"|"}" {printf("<%s,->",yytext);}
{ide} {printf("<IDE,%s>", yytext);}
{separatore} {printf(" ");}
{bin}|"-" {printf("<OP2,%s>", yytext);}



** Per saperne di piu' ***********************************************************
**      [A.V. Aho, R. Sethi and J.D. Ullman] pag. 105-113                            **
**      Neiemann T. A compact guide to Lex & Yacc, http://www.epaperpress.com        **
**      Flex and Bison, http://www.gnu.org                                           **
```

```
******************************************************************************************/
```