```
/*****************************************************************************
**
**                          Applicazioni in YACC
**
**      corso di Linguaggi Formali e Compilatori - Informatica quinquennale
**      corso di Compilatori - Informatica triennale
**
**              Marco Bellia - dip. Informatica - Univ. Pisa
**
**
** contiene:
**   1) Esercizio 5.4 in [AHO] pag.336 - Soluzione in Yacc consistente in
**      un parser e un esecutore per la riduzione di parentesi ridondanti in espres-
**      sioni con somma e prodotto. Associatività sinistra e precedenza di prodotto.
**   2) La soluzione Yacc si compone dei seguenti sorgenti:
**          Exp.lex - Linguaggio lessicale utilizzato per i simboli
**          ExpCompact.yacc - Grammatica attributata con linguaggio C come meta
**          structuresCompact.h - File C contenente strutture e procedure di supporto
**   3) Grammatica attributata LALR oblivious (in stile tradizionale) con marcatori:
**          8 terminals, 7 nonterminals
**          10 grammar rules, 16 states
**
*****************************************************************************/

%{
#include <stdlib.h>
#include <stdio.h>
#include "structuresCompact.h"
%}

%union{
    int dval;
    char *lex;
    struct coppia *dueAttributi;
};

%token <dval> NUM
%token <lex> IDE

%type <dueAttributi> Exp Fact Term
%type <dval> M1 M2


%%
Prog: Exp   {printf("= %s\n", $1->term);}
    ;

Exp: Exp '+' Fact   {char c = $1->opt;
                     if (c=='u') c='+';
                     $$ = newcoppia(c,concat1($1->term,concat1("+",$3->term)));
                     }
    | Fact      {$$ = $1;}
    ;

Fact: Fact '*' M1 Term  {char c = $1->opt;
                         if (c=='u') c='*';
                         $$ = newcoppia(c,concat1($1->term,concat1("*",$4->term)));
                         }
    | M2 Term   {$$ = $2;}
    ;
```

```
Term: IDE       {$$ = newcoppia('u',$1);}
    | '('Exp')' {if (($<dval>0==1)&&($2->opt=='+'))
                            $$ = newcoppia('+',concat1("(",concat1($2->term,")")));
                 else $$ = $2;
                 }
    ;

M1:     {$$ = 1;}
M2:     {$$ = 0;}

%%

/* COMPILIAMO:
bellia:EsempiComp marcobellia$ yacc -d ExpCompact.yacc
bellia:EsempiComp marcobellia$ lex Exp.lex
bellia:EsempiComp marcobellia$ cc -o ExpCompact.exe y.tab.c lex.yy.c -ly -ll

ESEGUIAMO:
bellia:EsempiComp marcobellia$ ~/Desktop/DIDATTICA/COMPILATORI/Materiale2012/Yacc2011/EsempiC
(x)
= x
bellia:EsempiComp marcobellia$ ~/Desktop/DIDATTICA/COMPILATORI/Materiale2012/Yacc2011/EsempiC
(x+y)+(y+z)
= x+y+y+z
bellia:EsempiComp marcobellia$ ~/Desktop/DIDATTICA/COMPILATORI/Materiale2012/Yacc2011/EsempiC
((a*(b+c))*(d))
= a*(b+c)*d
*/
```