Assignment #2

LR Expression Parser

Write an LR parser for Pascal Junior expressions. The parser, express, should use your lexical analyzer to read tokens, and parsing tables (pjtables) to guide parsing activities. In the event you need to modify the grammar provided, the file grammar.pj is provided in a format acceptable to the LR analyzer (lranal / lranal.p). These files are included in the EXPR.tar.gz file now on the class web site.

Print out each reduction as it is performed (LHS token -> RHS tokens). Use a compiler option (p+/-) to control printing reductions. For this assignment set the default to "true". Later you can set the default to "false". The program structure is:

```
+--------+       +--------+       +--------+
| driver |       | express |       | lexical |
|        |       |          |       | analyzer |
/+--------+       +--------+       +--------+  
| grammar :-----> |        |          |        |          |
|          |        |          |        |          |
/'''''''''*       */'''''''''*       */'''''''''*       */'''''''''*  
: grammar :-----+ :        :        :        :        :  
```

The expression parser should stop processing on any token that is not part of an expression (the driver will pull the non-expr token off the input stream). The above mentioned files and test data (sp*.inp) will be placed on the class web site. The data files will be made up of several expressions terminated by non-expr symbols. For example, you should be able to parse a file like:

```
a + b ;
(a < b) or (c < d);
(this or that) mod (old) ;
(* not really hard *) 3.1459+2e10-(a*b);
(chr(5)='c')or(ord('c')=2)and('zero'='a);
a*b ) (* argument in a procedure call *)
```

You will also be expected to recover gracefully from error conditions. Actions taken during error recovery will include a discard/flush of Stack/Input information. Printing of the information discarded/flushed by the expression parser is to be under compiler option control also (f+/-). For this assignment set the default value for f to be "true".
SLR Grammar for Pascal Junior

-- productions --

1 start  -->  expr
2 expr   -->  expr  relop  primary
3 expr   -->  primary
4 primary -->  primary  addop  term
5 primary -->  term
6 term   -->  term  multop  factor
7 term   -->  factor
8 factor -->  array
9 factor -->  builtin
10 factor -->  constant
11 factor -->  id
12 factor -->  fileid
13 factor -->  not  factor
14 factor -->  -  factor
15 factor -->  +  factor
16 factor -->  (  expr  )
17 builtin -->  id  (  expr  )
18 array  -->  id  [  expr  ]
19 relop  -->  <
20 relop  -->  <=
21 relop  -->  =
22 relop  -->  >=
23 relop  -->  >
24 relop  -->  <>
25 addop  -->  +
26 addop  -->  -
27 addop  -->  or
28 multop -->  *
29 multop -->  /
30 multop -->  div
31 multop -->  mod
32 multop -->  and
33 constant -->  realcon
34 constant -->  intgrcon
35 constant -->  boolcon
36 constant -->  alfacon
37 constant -->  charcon
38 constant -->  stringcon
39 boolcon -->  true
40 boolcon -->  false
41 s'    -->  start
-- terminals --

1 +
2 -
3 *
4 /
5 div
6 mod
7 and
8 or
9 <
10 <=
11 ==
12 >=
13 >
14 <>
15 not
16 fileid
17 id
18 ( 
19 )
20 realcon
21 intgrcon
22 alfacon
23 charcon
24 stringcon
25 true
26 false
27 [ 
28 ]
29 $ 

-- nonterminals --

30 start
31 expr
32 primary
33 term
34 factor
35 builtin
36 relop
37 addop
38 multop
39 constant
40 boolcon
41 array
42 s'
prodlhstable : array [ 1 ..  41 ] of integer;

prodlhstable[1] := 30;
prodlhstable[3] := 31;
prodlhstable[6] := 33;
prodlhstable[7] := 33;
prodlhstable[8] := 34;
prodlhstable[9] := 34;
prodlhstable[10] := 34;

prodlentable : array [ 1 ..  41 ] of integer;

prodlentable[1] := 1;
prodlentable[2] := 3;
prodlentable[3] := 1;
prodlentable[4] := 3;
prodlentable[5] := 1;
prodlentable[6] := 3;
prodlentable[7] := 1;
prodlentable[8] := 1;
prodlentable[9] := 1;
prodlentable[10] := 1;
### slr (1) Parse Table

This table consists of the action part of the parse table.

```plaintext
action : array [1 .. 54, 1 .. 29] of record
  act : 1..4;
  num : integer;
end;
```

*action* is an array of *(act, num)* pairs. It is accessed by *curr_state x terminal pair*

- **for the act entry:**
  - 1 => shift, 2 => reduce, 3 => error, 4 => accept

- **for the num entry:**
  - If *act* indicates a shift then *num* represents the state being moved to (i.e. the next state).
  - If *act* indicates a reduce then *num* represents the number of the production being used for the reduction.

```plaintext
action [1, 1].act := 1;
action [1, 1].num := 12;
action [1, 2].act := 1;
action [1, 2].num := 13;
action [1, 3].act := 3;
action [1, 3].num := 0;
action [1, 4].act := 3;
action [1, 4].num := 0;
action [1, 5].act := 3;
action [1, 5].num := 0;
action [1, 6].act := 3;
action [1, 6].num := 0;
action [1, 7].act := 3;
action [1, 7].num := 0;
action [1, 8].act := 3;
action [1, 8].num := 0;
action [1, 9].act := 3;
action [1, 9].num := 0;
action [1, 10].act := 3;
action [1, 10].num := 0;
action [1, 11].act := 3;
action [1, 11].num := 0;
action [1, 12].act := 3;
action [1, 12].num := 0;
action [1, 13].act := 3;
action [1, 13].num := 0;
action [1, 14].act := 3;
action [1, 14].num := 0;
```
**slr (1) parse table**

This table consists of the goto part of the parse table

\[
gotox : \text{array} \ [\ 1..54, \ 30..41\ ] \ \text{of integer}
\]

\(\text{gotox}\) is an array of integers. It is accessed by uncovered_state x non_terminal pairs.

Each array value represents the next state to go to, a value of a -1 indicates an error.

\[
gotox[\ 1, \ 30] := 23;
gotox[\ 1, \ 31] := 22;
gotox[\ 1, \ 32] := 21;
gotox[\ 1, \ 33] := 20;
gotox[\ 1, \ 34] := 19;
gotox[\ 1, \ 35] := 17;
gotox[\ 1, \ 36] := -1;
gotox[\ 1, \ 37] := -1;
gotox[\ 1, \ 38] := -1;
gotox[\ 1, \ 39] := 16;
gotox[\ 1, \ 40] := 7;
gotox[\ 1, \ 41] := 18;
gotox[\ 2, \ 30] := -1;
gotox[\ 2, \ 31] := -1;
gotox[\ 2, \ 32] := -1;
gotox[\ 2, \ 33] := -1;
gotox[\ 2, \ 34] := -1;
gotox[\ 2, \ 35] := -1;
gotox[\ 2, \ 36] := -1;
gotox[\ 2, \ 37] := -1;
gotox[\ 2, \ 38] := -1;
gotox[\ 2, \ 39] := -1;
gotox[\ 2, \ 40] := -1;
gotox[\ 2, \ 41] := -1;
gotox[\ 3, \ 30] := -1;
gotox[\ 3, \ 31] := -1;
gotox[\ 3, \ 32] := -1;
gotox[\ 3, \ 33] := -1;
gotox[\ 3, \ 34] := -1;
gotox[\ 3, \ 35] := -1;
gotox[\ 3, \ 36] := -1;
gotox[\ 3, \ 37] := -1;
gotox[\ 3, \ 38] := -1;
gotox[\ 3, \ 39] := -1;
gotox[\ 3, \ 40] := -1;
gotox[\ 3, \ 41] := -1;
gotox[\ 4, \ 30] := -1;
gotox[\ 4, \ 31] := -1;
\]