CMSC 124 Challenge Set #1

“Rather fail with honor than succeed by fraud.” - Sophocles

By Reinald Adrian DL. Pugoy

Instructions

- Answer only 3 problems below. Remember the things that we discussed in our lecture classes.
- Submit hardcopy, containing all your answers, on or before Thursday, September 3, 2009, 5:30 pm, C-114.
- Anyone who will submit after the deadline will get deductions.
- Trying to answer all of these will give a ‘feel of your second long exam’.
  Do your best. God bless. ☺

1. A certain programmer, named Mr. Guy Nerdilukin has designed a programming language called Driew Yllaer (or DY). When he designed DY, here were some tokens he invented:

TOKEN	**SAMPLE LEXEME**	**DESCRIPTION**
PRETTY_IDENT	12, a, b, hisgrade	A sequence of alphanumeric characters.
LITERAL	&1, &1000, &26	An ampersand sign followed by any combination of numeric constants.
ASSIGN_OP	#, =	Either the sharp sign or equal sign.

   Mr. Guy’s design was indeed weird, but you, as a caring individual decided to eventually adopt his conventions. Now your tasks are:

   a.) Site a valid statement which uses the three tokens.

   b.) Design transition diagrams for those tokens.

2. Given this grammar rule: **<factor> ::= id | {(<expr>)}**
   Use recursive-descent parsing algorithm to implement the above rule. Show the complete code. Also, provide facility for error-checking.

3. What will be different in the implementations of the following two declarations:

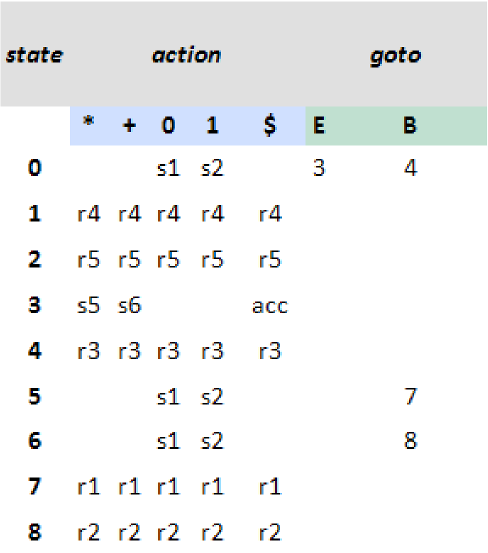
```
DECLARE PAY FIXED DECIMAL (7,2)
DECLARE PAY FIXED DECIMAL (7,4)
```

4. Given the following rough Pascal-like code:

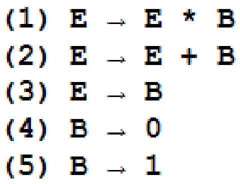
```
type mini = (1, 3, 5, 7);
var name: array[1..161] of integer;

function will_i_pass(n: integer, m: mini) : boolean  {
     if name[n] > m then
          will_i_pass = true;
     else
          will_i_pass = false;
}
```

   a) List all binding occurrences (considerations), and indicate the time they were performed.
   b) Show the set of values of those in **bold** using formal set notations.

5. Given ACTION and GOTO tables, use the LR parsing algo to parse the input **1 * 0 * 1 * 0 * 1**. Show the complete parsing table.



state	*action*					*goto*	
	*****	**+**	**0**	**1**	**$**	**E**	**B**
0			s1	s2		3	4
1	r4	r4	r4	r4	r4		
2	r5	r5	r5	r5	r5		
3	s5	s6			acc		
4	r3	r3	r3	r3	r3		
5			s1	s2			7
6			s1	s2			8
7	r1	r1	r1	r1	r1		
8	r2	r2	r2	r2	r2		



```
(1) E → E * B
(2) E → E + B
(3) E → B
(4) B → 0
(5) B → 1
```