Programming Assignment 3,

Due: Oct. 31, (Wednesday)

Consider the following grammar and its LL(1) parsing table for simple expressions:

1. $E \longrightarrow TE'$
2. $E' \longrightarrow +TE'$
3. $E' \longrightarrow \lambda$
4. $T \longrightarrow FT'$
5. $T' \longrightarrow *FT'$
6. $T' \longrightarrow \lambda$
7. $F \longrightarrow (E)$
8. $F \longrightarrow n$

LL(1) Parsing Table:

	n	$+$	$*$	(	)	\$
E	1			1		
E'		2			3	3
T	4			4		
T'		6	5		6	6
F	8			7		

Let n be any positive integer less than 32767.
Let \$ denote the end of the expression.

Implement a Recursive-descent parser according to the LL(1) parsing table above. Your parser only have to indicate whether or not the input expression is valid.

Requirements:

1. The user doesn't have to put \$ at the end of the expression. Your programs should prepare the token string with \$ attached automatically.
2. I will test your program on Unix using command line to compile and run your program. Name your program as LL1 (with appropriate extension). Write all your functions and modules in one file (i.e., LL1; so I don't have to link them)
3. If it is a java program, for example, after compiled, I will run your program as

   ```
   java LL1 123+((23+2*3)))+3
   ```

   The program should return some message to indicate whether or not it is a valid express.
4. If your program is not in C++ or Java, you have to describe how to compile and run your programs on my unix account. (You must use the languages with compiler available on our school's unix machine).
5. Follow the instructions for submission (read the web page of the class).