Christian Mesina
Hugo Romero
Luis Escobar Urrutia

Lab 5

Introduction:

- For this lab, we referenced the program from Lab 4 and used it to create a program that constructs the ACTION and GOTO parse tables for an SLR parser. We also constructed the SLR parse tables for the augmented expression grammar:
  S->E
  E->E+T
  E->T
  T->T*F
  T->F
  F->i
  F->(E)
  The input format should be the same as for Lab 4.

Questions:

1. Is the given expression grammar in SLR? (Note that if there are shift-reduce or reduce-reduce conflicts, the answer is NO.)
   - The given expression grammar is in SLR because there are no shift-reduce or reduce-reduce conflicts. This is because each state is occupied by AT MOST one only.

2. If there were shift-reduce conflicts, your program should set the state where the conflict occurs to SHIFT - this is what YACC does, and it works much of the time. Will this default action be correct in this case (assuming the grammar was not SLR)?
   - Yes, it should shift out of the conflict so that the program continues and this program in particular continues parsing through the SLR.

ACTION and GOTO Tables:

							go-to		
State	i	+	*	(	)	$	E	T	F
0	s5			s4			1	2	3
1		s6				acc			
2		r2	r7		r2	r2			
3		r4	r4		r4	r4			
4	s5			s4			8	2	3
5		r6	r6		r6	r6			
6	s5			s4				9	3
7	s5			s4					10
8		r6			s11				
9		r1	s7		r1	r1			
10		r3	r3		r3	r3			
11		r5	r5		r5	r5			

- The states are I_0, I_1-I_{11}

ACTION and GOTO Diagrams: