University of Nevada, Las Vegas Computer Science 456/656 Fall 2020

Answers to Assignment 5: Due Monday November 30, 2020

Your answers must be written in a pdf file and emailed to the graudate assistant, Shekhar Singh `shekhar.singh@unlv.edu` by 23:59 November 30. Your file must not exceed 10 megabytes, and must print out to at most 8 pages.

These problems refer to the handout titled "Simple LALR Parsers."

1. Answer the question asked after the tables in Example 1.

   The entry in row 6 column "e" of the action table is s7. That action forces the "e" to be combined into the same handle as the nearest "i"

2. In this question, refer to Example 2. What follows is the computation if the input is $x - x - x$. Answer the question given after the tables.

$\$_0$	$x - x - x\$$	
$\$_0 x_2$	$-x - x\$$	
$\$_0 E_1$	$-x - x\$$	1
$\$_0 E_1 -_3$	$x - x\$$	1
$\$_0 E_1 -_3 x_2$	$-x\$$	1
$\$_0 E_1 -_3 E_4$	$-x\$$	11
$\$_0 E_1$	$-x\$$	112
$\$_0 E_1 -_3$	$x\$$	112
$\$_0 E_1 -_3 x_2$	$\$$	112
$\$_0 E_1 -_3 E_4$	$\$$	1121
$\$_0 E_1$	$\$$	11212

   halt

   The entry in row 4 column "$-$" of the action table is r2. That forces subtraction to be left-associative.

3. Answer the question given after the tables in Example 4.

   The entry in row 4 column "$\wedge$" of the action table causes "E" on top of the stack to be combined with the following "$\wedge$" instead of the previous "$\wedge$" forcing the operator to be right-associative. If the operator were left-associative, the entry would be "r2"

4. Answer the question given after the tables in Example 5.

   Row 4, column "$*$." If the entry were "r2" the operators would have equal precedence.

5. Refer to Example 6. Write a computation of that parser with the input $x - (x - x)$.

$\$_0$	$x - (x - x)\$$	
$\$_0 x_2$	$-(x - x)\$$	
$\$_0 E_1$	$-(x - x)\$$	1
$\$_0 E_1 -_3$	$(x - x)\$$	1
$\$_0 E_1 -_3 (_5$	$x - x)\$$	1
$\$_0 E_1 -_3 (_5 x_2$	$-x)\$$	1
$\$_0 E_1 -_3 (_5 E_4$	$-x)\$$	11
$\$_0 E_1 -_3 (_5 E_4 -_3$	$x)\$$	11
$\$_0 E_1 -_3 (_5 E_6 -_3 x_2$	$)\$$	11
$\$_0 E_1 -_3 (_5 E_6 -_3 x_2$	$)\$$	11
$\$_0 E_1 -_3 (_5 E_6 -_3 E_4$	$)\$$	111
$\$_0 E_1 -_3 (_5 E_6 -_3 E_6)_7$	$\$$	111
$\$_0 E_1 -_3 E_4$	$\$$	1113
$\$_0 E_1$	$\$$	11132

halt

6. Refer to Example 3. According to your high school algebra teacher (YHSAT), the expression $-x - y$ is evaluted by first computing the additive inverse of the value of x, then subracting the value of y. That corresponds to the precedence determined by Example 3. On the other hand, YHSAT would tell you that the value of $-x * y$ is the additive inverse of x times y, but C++ would compute the product of $-x$ and y. You might think that it doesn't matter, but it does. Operators can be overloaded in some programming languages as well as in the literature, and $(-x) * y$ might not equal $-(x * y)$.

   If you change Example 3, replaceing subtraction with multiplication, your parser will be consistent with C++. But what if, instead, you want to please YHSAT? Using Example 3 as a guide, define an LALR parser for the grammar below which is consistent with what YHSAT wants.

   1. $E \to x$
   2. $E \to E * E$
   3. $E \to -E$

   We first rewrite the grammar with stack states:

   1. $E \to x_2$
   2. $E \to E *_3 E_4$
   3. $E \to -_5 E_6$

   Here are the ACTION and GOTO tables, if we retain the precedences of C++:

	x	$-$	$*$	\$	E
0	$s2$	$s5$			1
1			$s3$	**halt**	
2			$r1$	$r1$	
3	$s2$	$s5$			4
4			$r2$	$r2$	
5	$s2$	$s5$			6
6			$r3$	$r3$	

   We cannot please YHSAT, since the grammar generates strings like $- - x$ and $x * -x$. However, we can at least resolve the ambiguity of $-x * x$ in favor of YHSAT, as follows:

	x	$-$	$*$	\$	E
0	$s2$	$s5$			1
1			$s3$	**halt**	
2			$r1$	$r1$	
3	$s2$	$s5$			4
4			$r2$	$r2$	
5	$s2$	$s5$			6
6			$s3$	$r3$	