Homework #4

______________________ Name ____ Sec

Questions:	Answers:
1. Given the following grammar: <expr> ::= <term> | <expr> <op1> <term> <term> ::= <darg> | <term> <op2> <darg> <darg> ::= <digit> | <darg> <digit> <digit> ::= 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 <op1> ::= + | - <op2> ::= * | / Extend this grammar to include the comparison operators, =, <=, >=, <, and >, which are all at the same level of precedence. Their precedence is below that of + and -. Your new grammar must also be unambiguous.	
2. Working from the grammar you extended in the previous problem, further extend the expression grammar to include parentheses and the unary minus sign. Your new grammar must also be unambiguous. Notes: (1) The unary minus has a higher precedence than the other operators; for instance, –2*–3 is grouped (–2)*(–3). (2) As usual, parentheses dictate that parenthesized expressions execute first. (3) The unary minus applies to parenthesized expressions as well as to individual numbers. Thus, – (2*3+4) is valid, as is –2*3. (4) Expressions like – – –2 are also valid.	

3. Reduce the following grammar by crossing out rules that are “silly,” “unproductive,” or “unreachable” as defined in the class lecture notes. Label each crossed out rule appropriately with one of these three designations. The start symbol is S. $S \rightarrow bS$ $S \rightarrow ba$ $S \rightarrow A$ $S \rightarrow S$ $A \rightarrow a$ $A \rightarrow Bb$ $B \rightarrow C$ $B \rightarrow ba$ $B \rightarrow aCC$ $C \rightarrow aD$ $C \rightarrow D$ $D \rightarrow Df$ $G \rightarrow Ha$ $G \rightarrow a$ $H \rightarrow Ga$	
4. The following grammar is not a simple LL(1) grammar. Without changing the language, modify the grammar so that it is simple LL(1). $S \rightarrow ABAe$ $A \rightarrow dB \mid aS \mid c$ $B \rightarrow AS \mid b$	
5. For the grammar in Problem 4, do the following. a) Compute FIRST(ABAe). b) Compute FIRST(AS) $\cap$ FIRST(b) c) Although the grammar in Problem 4 is not simple LL(1), why is it nevertheless LL(1)?	

6. Give a parse table for the following grammar. (S is the start symbol.)

1. S →ABe
2. A →dB
3. A →aS
4. A →c
5. B →AS
6. B →b

```
_| a    | b | c    | d    | e |
S|ABe|    |ABe|ABe|    |
A|aS  |    | c    |dB   |    |
B|AS  | b |AS  |AS  |    |
```

7. Using your parse table in Problem 6, give a trace of the parse for the input string dbbe. Give the unused input string, stack, and output (sequence of rule numbers) at the beginning of each iteration. (The answer should look similar to the traces we did in class.)

Stack:	Input String:	
S#	dbbe#	<- S->ABe
ABe#	dbbe#	<- A->db
dbBe#	dbbe#	<- Pop off matching 'd' at beginning
bBe#	bbe#	<- Pop off the matching 'b's...
Be#	be#	<- B->b
be#	be#	<- Pop off 'b's...
e#	e#	<- Pop off 'e's...
#	#	<- Means the input was valid