Homework 5 - LR

Due Date: Tuesday, November 22, 2005, 2:00

Your Name: ________________________________________

1. Can a non-recursive predictive parser handle all LR(1) grammars?

________________

2. What sort of derivation does a bottom-up parser discover?

________________

3. In bottom-up parsing, at each step, we are looking for a what? (One word.)

________________

4. What are the four possible actions in a shift-reduce parser?

________________

________________

________________

________________

5. In a shift action, what happens?

___________________________________________

___________________________________________

6. In a reduce action, assume we are reducing by the rule

\[ S \rightarrow a \ S \ b \ C \]

Show the first few symbols on top of the stack. (Assume we are using shift-reduce parsing, in general, so you don’t need to show any states. Please draw your stack so that the top of the stack is toward the top of the page.)
7. The set of grammars that can be parsed with the LR algorithm is
   _____ a proper subset of
   _____ a proper superset of
   _____ equal to
   _____ unrelated to
   the set of grammars parsable with predictive parsing. (Please check the correct relation.)

8. Can an LR grammar be ambiguous? __________

9. In the LR parsing algorithm, what kinds of things are pushed onto the stack?
   __________________________________________________________________________

10. In LR parsing, what are the two tables called?
    __________________________________________________________________________

11. Here is a grammar; give the set of all LR(0) items for this grammar.
    \[ S' \rightarrow S \]
    \[ S \rightarrow ( A ) | \varepsilon \]
    \[ A \rightarrow S , A | S \]
    ____________ ____________ ____________
    ____________ ____________ ____________
    ____________ ____________ ____________
    ____________ ____________ ____________
    ____________ ____________ ____________

12. What algorithm does YACC use? ____________

13. What is the input to YACC? ____________
What is the output? ____________

14. What is the difference between an LR(0) and an LR(1) item?
    __________________________________________________________________________
15. Here is a grammar:
   \[ S \rightarrow b \ T \ c \nonumber \]
   \[ \rightarrow \ S \ T \ S \ \mathbf{g} \nonumber \]
   \[ \rightarrow \ T \ d \ S \ \mathbf{h} \nonumber \]
   \[ \rightarrow \varepsilon \nonumber \]
   \[ T \rightarrow a \ T \nonumber \]
   \[ \rightarrow \varepsilon \nonumber \]

   What is FIRST(S)? _______________________
   What is FIRST(T)? _______________________
   What is FOLLOW(S)? _______________________
   What is FOLLOW(T)? _______________________

16. In bottom-up parsing, we look for a ____________________________________________
    and when we find one, we reduce. Otherwise, we ____________________________.

17. Here is a grammar:
   \[ S' \rightarrow E \nonumber \]
   \[ E \rightarrow E + T \nonumber \]
   \[ \rightarrow T \nonumber \]
   \[ T \rightarrow T * F \nonumber \]
   \[ \rightarrow F \nonumber \]
   \[ F \rightarrow (E) \nonumber \]
   \[ \rightarrow \mathbf{id} \nonumber \]

   List all the LR(1) items in CLOSURE ( \{ S' \rightarrow \varepsilon , ( \} )

   ____________________________ ____________________________ ____________________________
   ____________________________ ____________________________ ____________________________
   ____________________________ ____________________________ ____________________________
   ____________________________ ____________________________ ____________________________
   ____________________________ ____________________________ ____________________________
   ____________________________ ____________________________ ____________________________
   ____________________________ ____________________________ ____________________________
   ____________________________ ____________________________ ____________________________

18. Using the grammar from the previous question...
    Let \( CC_1 = \{ E \rightarrow E \cdot + T , ( \)
    \[ E \rightarrow T \cdot , ( \} \)

    What is GOTO (CC_1, +)?
    ____________________________ ____________________________ ____________________________