University of Toronto
Department of Electrical and Computer Engineering

Midterm Examination

CSC467 Compilers and Interpreters
Fall Semester, 2005

Time and date: TBA
Location: TBA

Print your name and ID number neatly in the space provided below; print your name at the upper right corner of every page. Place a picture ID on your table for verification.

Name:
ID Number:

This booklet should be 6 pages long including this page. If not, report this to the instructor or the TA. Do all problems in this booklet. Try not to spend too much time on any one question.

This is an open book exam. You are allowed to refer to the text and your notes. **Do all work in the space provided**. You can use the backs of sheets as scrap paper, if necessary. Ask the proctor if you need more paper. **Nothing in the back of the sheets will be graded**.

Question	Points	Score	Grader
1	40		
2	25		
3	35		
Total	100		

1. **(Short Answers, 40 points)**

I. (5 points) True or false? Are the following regular expressions exactly equivalent? The special symbol ϵ represents the empty string.

a)	$x?x^*$	x^*	True	False
b)	$y^*\|z^*$	$(y\|z)^*$	True	False
c)	a^*b^*	$(ab)^*$	True	False
d)	$(P\|Q\|\epsilon)^*$	$(P\|Q)^*$	True	False
e)	$(0\|1)?$	$0?\|1?$	True	False

Answers: True, False, False, True, True.

II. (Project Question, 9 points) A class of shift/reduce conflicts Bison can generate are due to its limited lookahead. For example:

```
rule: command optional_keyword ’(’ identifier_list ’)’
    ;

optional_keyword: /* blank */
                | ’(’ keyword ’)’
                ;
```

Can you use precedence rules to resolve this conflict ? If not, are there other ways of resolving this conflict without changing the syntax of the command language being accepted?

Answer: Bison’s operator precedence rules cannot be used to resolve the conflict. The problem is that two similar but distinct productions share a prefix; in this case the distinct productions are implicit rather than explicit (that is, a keyword in brackets may or may not come between a command and an identifier list, which is also in brackets). A simple solution is to use an intermediate rule that contains the prefix, then non-terminal(s) that distinguish the various productions, e.g.

```
rule: command intermediate_rule
    ;

intermediate_rule: ’(’ id_or_keyandid ’)’
                 ;

id_or_keyandid: identifier_list
              | keyword ’)’ ’(’ identifier_list
              ;
```

III. (26 points) True or False? Each of these True/False questions is worth 2 points. Mark one entry. If we cannot understand what entry you marked, you may miss credit. *Warning!* You will recieve −1 points for any wrong answers.

1. Lexical analysis is recursive in order to handle nested parentheses.
   ( ) TRUE (X) FALSE

2. Scanners don't know anything about the grammar of a language.
   (X) TRUE ( ) FALSE

3. A successful parse means the input is semantically correct.
   ( ) TRUE (X) FALSE

4. Finite State Machines can have an unlimited number of states.
   ( ) TRUE (X) FALSE

5. A regular expression is a type of pattern used to classify lexemes.
   (X) TRUE ( ) FALSE

6. You can change state in a DFA without reading any input character.
   ( ) TRUE (X) FALSE

7. Transition Tables are indexed with current state and next state.
   ( ) TRUE (X) FALSE

8. Syntax analysis handles type checking and type conversions, e.g. int to float.
   ( ) TRUE (X) FALSE

9. Regular expressions cannot be used to match strings of balanced parentheses.
   (X) TRUE ( ) FALSE

10. $0(00)^*$ is a regular expression that matches only non-empty strings containing an odd number of zeroes.
    (X) TRUE ( ) FALSE

11. $(000)^*$ is a regular expression that matches only strings containing an odd number of zeroes, including the empty string.
    ( ) TRUE (X) FALSE

12. Finite State Machines can have only one edge leaving the same state labelled with the same label (character).
    ( ) TRUE (X) FALSE

13. A DFA must have exactly one final (accepting) state.
    ( ) TRUE (X) FALSE

2. **(Syntax-Directed Translation, 25 points)** Write an attribute grammar for the floating point value of a decimal number given by the following grammar. (Hint: Use a *count* attribute to count the number of digits to the right of the decimal point.)

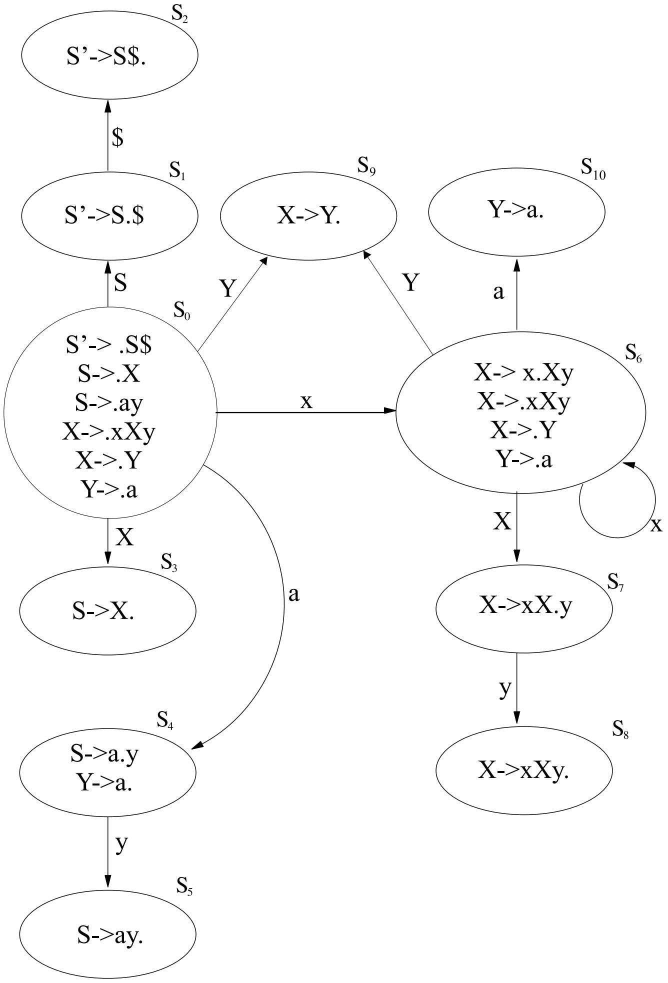
$$\begin{aligned} dnum &\rightarrow num.num \\ num &\rightarrow num\ \ digit \mid digit \\ digit &\rightarrow 0 \mid 1 \mid 2 \mid 3 \mid 4 \mid 5 \mid 6 \mid 7 \mid 8 \mid 9 \end{aligned}$$

PRODUCTION	SEMANTIC RULES
$dnum \rightarrow num_1.num_2$	$dnum.val := num_1.val + num_2.val/10^{num_2.count}$
$num \rightarrow num_1\ \ digit$	$num.val := 10 \times num_1.val + digit.val$ $num.count := num_1.count + 1$
$num \rightarrow digit$	$num.val := digit.val$ $num.count := 1$
$digit \rightarrow 0$	$digit.val := 0$
$digit \rightarrow 1$	$digit.val := 1$
$digit \rightarrow 2$	$digit.val := 2$
. . .	
$digit \rightarrow 8$	$digit.val := 8$
$digit \rightarrow 9$	$digit.val := 9$

3. **(Parsing, 35 points)** Consider the following grammar.

$$\begin{aligned} S &\rightarrow X \mid ay \\ X &\rightarrow xXy \mid Y \\ Y &\rightarrow a \end{aligned}$$

(a) Compute the set of SLR(1) items for this grammar and then complete the corresponding DFA below. Show the contents of each state (set of items) and label the transitions clearly. (20 points)



(b) Is this grammar SLR(1)? Briefly explain why or why not. (5 points)

Answer: `The grammar is not SLR(1); there is a SHIFT-REDUCE conflict in state` S_2`.`

(c) Compute the FIRST sets for x and X, and the FOLLOW set for X. (10 points)

Answer:

$$\begin{aligned} \text{FIRST}(x) &= \{x\} \\ \text{FIRST}(X) &= \{x, a\} \\ \text{FOLLOW}(X) &= \{y, \$\} \end{aligned}$$