

1. **Regular Languages (15 pts)** Give a DFA that recognizes the same language as the RE below. Remember that the + operator means one or more.

$$(a \mid b)^+ a b b (a \mid b)^+$$

2. **First/Follow (15 pts)** For each grammar below, give the first and follow information for the non-terminals. Which grammars are LL(1)?

- A. $S \rightarrow A B c$
 $A \rightarrow a$
 $A \rightarrow \epsilon$
 $B \rightarrow b$
 $B \rightarrow \epsilon$

	First	Follow
S		
A		
B		

Grammar LL(1)?

- B. $S \rightarrow A b$
 $A \rightarrow a$
 $A \rightarrow B S$
 $A \rightarrow \epsilon$
 $B \rightarrow x$
 $B \rightarrow \epsilon$

	First	Follow
S		
A		
B		

Grammar LL(1)?

- C. $S \rightarrow a S e$
 $S \rightarrow B$
 $B \rightarrow b B f$
 $B \rightarrow C$
 $C \rightarrow c C g$
 $C \rightarrow d$

	First	Follow
S		
B		
C		

Grammar LL(1)?

2. **Lex (10 pts)** For the lex specification given below, what is the output if the string `abbbaabcaaaaaaebbbbbaaabbbaa` is the input?

```

a      { /* print out a '*' */ }
aa     { /* print out first character in matched string */ }
a+    ;
b      { /* print out a '*' */ }
bbb   { /* print out first character in matched string */ }
b+    ;
.     ;

```

3. **LL parsing (10 pts)** Consider this recursive descent parser. Assume `match()` is defined as given in class. What does the associated grammar look like? What is the input alphabet?

```

S() {
    if (lookahead == 'a') { match('a'); S(); match('b'); }
    else if (lookahead == 'x' or lookahead == 'y') { A(); }
    else error();
}

```

```

A() {
    if (lookahead == 'x') { match('x'); A(); }
    else if (lookahead == 'y') { match('y'); B(); }
    else error();
}

```

```

B() {
    if (lookahead == 'z') { match('z'); B(); match('c'); }
    else if (lookahead == '$ or lookahead == 'b') return;
    else error();
}

```

4. **LR parsing (15 pts)** Use the action/goto table for the grammar below (where the productions are as numbered) to parse the incorrect input string:

(2 ()) 5) 6)

When the syntax error is detected, a) what does the stack look like, 2) what was the last action and 3) what is the lookahead?

of your syntax but your meaning should be clear. You can assume the `int` token has a `val` attribute that holds the integer value of the lexeme.

- Draw the parse tree for the input above and show how your attributes are used for the computation.

R → **S**

S → (**L**) **L**

S → ε

L → `int` **L**

L → **S**

6. Short answer (20 points) Choose **two** of the three below to answer. Use complete sentences and correct grammar. Be sure your answer is brief and to the point. Clearly indicate which question you are answering.

- What is a FIRST set and how is it used in the generation of predictive parse tables (and recursive descent parsers) for LL parsing?
- Explain the terms ‘shift/reduce conflict’ and ‘reduce/reduce conflict’ with respect to LR(1) parsing.
- Explain the tasks done by the lexical analysis phase and by the syntax analysis phase. How do these two phases work together to do language recognition?