1. (12 pts) Consider the following languages. Write “REG” if it is regular, “CFL” if it is a CFL and not regular, and write “NOT” if it is not a CFL.

   (a) \( L = \{a^mb^n c^n \mid n > 0, m > 0, m < n \} \), \( \Sigma = \{a, b, c\} \). \[\text{NOT}\]

   (b) \( L = \{b^na^nb^n \mid n > 0\} \), \( \Sigma = \{a, b\} \). \[\text{CFL}\]

   (c) \( L = \{a^nb^pc^p \mid n > 0, p > 0, n \text{ mod } 2 = 0\} \), \( \Sigma = \{a, b, c\} \). \[\text{CFL}\]

   (d) \( L = \{a^mb^nc^n \mid p > n > m > 200\} \), \( \Sigma = \{a, b, c\} \). \[\text{NOT}\]

   (e) \( L = \{w \in \Sigma^* \mid n_a(w) + n_b(w) < n_c(w)\} \), \( \Sigma = \{a, b, c\} \). \[\text{CFL}\]

   (f) \( L = \{b^ma^na^nb^n \mid m > 0, n > 0, p > 0\} \), \( \Sigma = \{a, b\} \). \[\text{REG}\]

2. (12 pts) Answer TRUE or FALSE to each of the statements below.

   (a) Removing \( \lambda \)-productions from a CFG \( G_1 \) can result in unit productions in the transformed and equivalent CFG \( G_2 \). \( \text{TRUE} \) or \( \text{FALSE} \)?

   (b) For FIRST, a function helpful in calculating parse tables, \( S \) is always in the FIRST(S) for start symbol \( S \). \( \text{TRUE} \) or \( \text{FALSE} \)?

   (c) If the marked productions \( A \rightarrow ab, cS \) and \( B \rightarrow ab \) are in the same state in the DFA that models the LR parsing process for a CFG, then there must be a shift-reduce conflict with that state. \( \text{TRUE} \) or \( \text{FALSE} \)?

   (d) If \( M_1 \) is an NPDA and \( M_2 \) is an NPDA, then there exists a TM \( M_3 \) such that \( L(M_3) = L(M_1) \cap L(M_2) \). \( \text{TRUE} \) or \( \text{FALSE} \)?

   (e) If TM \( M_1 \) is an acceptor, can read and write tape symbols, only moves the tape head right and halts on all inputs, then there exists a DFA \( M_2 \) such that \( L(M_1) = L(M_2) \). \( \text{TRUE} \) or \( \text{FALSE} \)?

   (f) If G is a CFG and \( M \) is an NPDA with only 3 states, then \( L(G) \cap L(M) \) is a CFL. \( \text{TRUE} \) or \( \text{FALSE} \)?
3. (3 pts) Give an example of a CFG that is not also a regular grammar.

\[
S \rightarrow aSb \\
S \rightarrow \lambda
\]

4. (3 pts) List the four types of actions in an LR parse table.

- Shift
- Reduce
- Accept
- Error

5. (4 pts) Consider the following CFG.

\[
S \rightarrow BCa \\
A \rightarrow aa \mid aaAB \\
B \rightarrow bB \mid \lambda \\
C \rightarrow Ac \mid \lambda
\]

(a) Show a derivation of a string in which c is immediately following B in a sentential form.

\[
S \Rightarrow BCa \Rightarrow BAcA \Rightarrow BaaA[B]ca \Rightarrow aaABca \\
\Rightarrow aaaa Bca \Rightarrow aaaaAca
\]

(b) Show a derivation of a string in which a is immediately following B in a sentential form.

\[
S \Rightarrow BCa \Rightarrow BAcA \Rightarrow [B]aaca \Rightarrow aaca
\]

6. (3 pts) The following grammar is LL(k) for what value of k? Give the value of k and an example of two strings that need that value of k to distinguish which rule to apply.

\[
S \rightarrow ABC \mid DABe \\
A \rightarrow abA \mid \lambda \\
B \rightarrow baC \mid bd \\
C \rightarrow aaC \mid ac \\
D \rightarrow bdAc \mid db
\]

\[
k = 4
\]

\[
S \Rightarrow bdac \\
S \Rightarrow bdab_bcbde
\]
7. (10 pts) Consider the following grammar (DO NOT change the grammar):

\[
\begin{align*}
S & \rightarrow BA \\
A & \rightarrow aB \mid \lambda \\
B & \rightarrow cBAd \mid b
\end{align*}
\]

For this problem you will construct the LL(1) parse table.

(a) Calculate FIRST and FOLLOW for the variables in the grammar.

<table>
<thead>
<tr>
<th></th>
<th>FIRST</th>
<th>FOLLOW</th>
</tr>
</thead>
<tbody>
<tr>
<td>S</td>
<td>c, b</td>
<td>$</td>
</tr>
<tr>
<td>A</td>
<td>a, \tau</td>
<td>d, $</td>
</tr>
<tr>
<td>B</td>
<td>c, b</td>
<td>a, d, $</td>
</tr>
</tbody>
</table>

(b) Calculate all entries in the LL(1) Parse Table. If there are multiple rules for an entry, write in all the rules.

<table>
<thead>
<tr>
<th></th>
<th>a</th>
<th>b</th>
<th>c</th>
<th>d</th>
<th>$</th>
</tr>
</thead>
<tbody>
<tr>
<td>S</td>
<td>BA</td>
<td>BA</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>A</td>
<td>aB</td>
<td></td>
<td>\tau</td>
<td>\tau</td>
<td></td>
</tr>
<tr>
<td>B</td>
<td>b</td>
<td>cBAd</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

(c) Is this grammar an LL(1) grammar? Explain. yes, no conflicts in the table.
8. (16 pts) Construct the LR parsing table for the following grammar (DO NOT change the grammar.) A new start symbol $S'$ and production have already been added to the grammar.

$0) S' \rightarrow S$
$1) S \rightarrow ABc$
$2) A \rightarrow a$
$3) A \rightarrow \lambda$
$4) B \rightarrow bB$
$5) B \rightarrow c$

(a) Calculate the FIRST and FOLLOW sets of variables.

<table>
<thead>
<tr>
<th></th>
<th>FIRST</th>
<th>FOLLOW</th>
</tr>
</thead>
<tbody>
<tr>
<td>$S$</td>
<td>$a, b, c$</td>
<td>$$</td>
</tr>
<tr>
<td>$A$</td>
<td>$a, \lambda$</td>
<td>$b, c$</td>
</tr>
<tr>
<td>$B$</td>
<td>$b, c$</td>
<td>$c$</td>
</tr>
</tbody>
</table>

(b) Construct the transition diagram of the DFA that models the stack. Number the states, show marked productions, and identify final states by two circles.
(c) Construct the LR parse table that corresponds to the transition diagram drawn in part b. (Note: all the rows and columns given may not be needed. If there are multiple items for an entry, write all in the entry.)

<table>
<thead>
<tr>
<th></th>
<th>a</th>
<th>b</th>
<th>c</th>
<th>$</th>
<th>A</th>
<th>B</th>
<th>S</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>s3</td>
<td>r3</td>
<td>r3</td>
<td>1</td>
<td>2</td>
<td></td>
<td></td>
</tr>
<tr>
<td>1</td>
<td></td>
<td>s5</td>
<td>s6</td>
<td>4</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>2</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>acc</td>
<td></td>
<td></td>
</tr>
<tr>
<td>3</td>
<td></td>
<td>r2</td>
<td>r2</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>4</td>
<td></td>
<td></td>
<td>s7</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>5</td>
<td></td>
<td>s5</td>
<td>s6</td>
<td>8</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>6</td>
<td></td>
<td></td>
<td></td>
<td>r5</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>7</td>
<td></td>
<td></td>
<td></td>
<td>r1</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>8</td>
<td></td>
<td></td>
<td></td>
<td>r4</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>9</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>10</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>11</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>12</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
9. (8 pts) Consider the following L-system.

Axiom: X f g
X → g [ + g g ] Y X - g
Y → g

angle 90
color black
lineWidth 2
distance 10

Recall that g is for drawing a line, f is for moving forward, + means change the direction by the angle clockwise, - means change the direction by the angle counterclockwise and [ ] are used for stacking operations.

Assume a g drawn with distance 10 and lineWidth 2 is about this size |

a. Render the L-system and draw the axiom if there is a visual picture for it.

```
 /  
```

b. Give the first string in the language (after the axiom) and draw it.

```
g[+gg] Y X - g f g
```

```
  - 
```

c. Give the second string in the language (after the axiom) and draw it.

```
g[+gg] g g [+gg] Y X - g - g f g
```

```
  /
```
10. (6 pts) **Pumping Lemma for CFL's** Let \( L \) be any infinite CFL. Then there is a constant \( m \) depending only on \( L \), such that for every string \( w \) in \( L \), with \( |w| \geq m \), we may partition \( w = uvxyz \) such that:

\[
|vxy| \leq m, \text{ (limit on size of substring)} \\
|vy| \geq 1, \text{ (\( v \) and \( y \) not both empty)} \\
\text{For all } i \geq 0, \ uv^i xy^i z \in L
\]

Consider \( L = \{a^n b^p c^q \mid q < n, q < p, q > 0\} \Sigma = \{a, b, c\} \).

Prove \( L \) is not a context-free language.

You only have to fill in the parts below. Assume \( L \) is a context-free language.

(a) Choose \( w = a^{m+1} b^{m+1} c^m \)

(b) Prove the case when \( v = a^{t_1} \) and \( y = a^{t_2} \)

\[
i = 0 \quad w = uv^0 xy^0 z = a^{m+1-t_1-t_2} b^{m+1} c^m \not\in L
\]

since \( n_a(w) \leq n_c(w) \)

so \( w^i \not\in L \)

(c) Prove the case when \( v = b^{t_1} \) and \( y = c^{t_2} \)

if \( t_2 > 0 \) let \( i = 2 \)

\[
w^i = uv^2 xy^2 z = a^{m+1} b^{m+t_1} c^{m+t_2}
\]

since \( n_a(w_i) \leq n_c(w_i) \) so \( w^i \not\in L \)

if \( t_2 = 0 \), let \( i = 0 \)

\[
w^i = uv^0 xy^0 z = a^{m+1} b^{m+1-t_1} c^m
\]

since \( n_b(w_i) \leq n_c(w_i) \) so \( w^i \not\in L \)
11. (10 pts) Construct a one-tape deterministic Turing machine transducer (using a transition diagram) that computes the following function:

\( w \in \{a, b\}^*, |w| \geq 1, f(w) = w' \) where \( w' \) is \( w \) with every occurrence of \( ab \) replaced by \( ba \).

For example, \( f(abaabb) = baabab \) with the two \( ab \)'s each replaced by a \( ba \) in the string. \( f(aaaabbabaa) = aaababaa \). \( f(aaa) = aaa \), since there are no \( b \)'s.

In drawing the transition diagram, remember to identify the start state by an arrow and final states by double circles. Format of labels are \( a; b, R \) where \( a \) is the symbol read on the tape, \( b \) is the symbol written to the tape and \( R \) is the direction moved (you can use \( L \) and \( R \) for directions.)

\( |w| = n \). What is the worst case running time (big-Oh) of your TM? \( O(n) \)
12. (10 pts) Construct a deterministic TM (using building blocks) that computes the following function:

\[ w \in \{a, b\}^*, |w| > 0, f(w) = y \text{ if } w \text{ has } n_a(w) = n_b(w), \text{ else } f(w) = n. \]

For example, \( f(ababaa) = n \) since the number of \( a \)'s is 4 which is not equal to the number of \( b \)'s, which is 2. \( f(aababb) = y, f(aaab) = n, \) and \( f(bbbbaa) = y. \)

See the building block notation on later pages. Make sure the tape head is pointing to the leftmost symbol of the output.

B is blank

\[
\begin{align*}
& \rightarrow L_B R_a | B \rightarrow A L_B R_b | B \rightarrow C \\
& \quad \downarrow B \quad \downarrow B \\
& \quad R_b | B \rightarrow R_B R_n h \quad R_B R_n h \\
& \quad \downarrow B \quad \downarrow B \\
& \quad R_B R_y h
\end{align*}
\]

Note the \( A + C \) are symbols I'm writing on the tape.

Assume \( |w| = n \). What is the running time in terms of \( n \) (big-Oh) of your TM? \( O(n^2) \)
Notation for Simplifying Turing Machines

Suppose \( \Gamma = \{a, b, c, B\} \)

- \( z \) is any symbol in \( \Gamma \)
- \( x \) is a specific symbol from \( \Gamma \)

1. \( s \) - start
2. \( R \) - move right
3. \( L \) - move left
4. \( x \) - write \( x \) (and don’t move)
5. \( R_a \) - move right until you see an \( a \) (note that this moves right at least one square before it checks for \( a \).)
6. \( L_a \) - move left until you see an \( a \)
7. \( R_{alb} \) - move right until you see an \( a \) or \( b \)
8. \( L_{alb} \) - move left until you see an \( a \) or \( b \)
9. \( R_{\neg a} \) - move right until you see anything that is not an \( a \)
10. \( L_{\neg a} \) - move left until you see anything that is not an \( a \)
11. \( h \) - halt in a final state
12. \( \{a, b\} \rightarrow w \)

If the current symbol is \( a \) or \( b \), let \( w \) represent the current symbol.

13. \( C \) - copy a string, \( F(w) = w0w \), makes a copy of the string and inserts a 0 between the original and the copy.
14. \( S_L \) - Shift the string that is to the right of the tape head (up to a blank), to the left, writing over the symbol the tape head is pointing to.
15. \( S_R \) - Shift the string that is to the left of the tape head (up to a blank), to the right, writing over the symbol the tape head is pointing to.