INSTRUCTIONS
Read the following carefully before answering any question.

- Make sure you have filled in your name and USB ID number in the space above.
- Write your answers in the space provided; Keep your answers brief and precise.
- The exam consists of 4 questions, in 9 pages (including this page) for a total of 30 points.
  Question 1 has two pages
  Question 2 is on a single page.
  Question 3 has three pages.
  Question 4 has two pages.

GOOD LUCK!
1. [Total: 8 points] Consider the following three languages:

$L_1$: strings over \{a, b, c\} that contain an even number of a’s. (Note: zero is an even number.)

$L_2$: strings over \{a, b, c\} that have ab as a substring.

Note: A substring of string s is obtained by deleting zero or more symbols from the beginning and end of s; for instance banana, nan and \epsilon are substrings of banana

$L_3$: strings over \{a, b, c\} that do not have ab as a substring.

a. (1 point) Write a regular expression corresponding to $L_1$.

b. (1 point) Write a regular expression corresponding to $L_2$.

c. (1.5 points) Draw a (possibly Nondeterministic) Finite Automaton that accepts $L_1$.

d. (1.5 points) Draw a (possibly Nondeterministic) Finite Automaton that accepts $L_2$.
f. (1.5 points) Write a regular expression corresponding to $L_3$.

e. (1.5 points) Draw a (possibly Nondeterministic) Finite Automaton that accepts $L_3$. 
2. [4 points]

Consider the following Lex specification:

digit [0-9]
alpha [a-zA-Z]
%%
"class" {return{CLASS};}
{alpha}{(alpha)|{digit}}* {return(ID);}
"return" {return(RETURN);}
{digit}+ {return{INT};}
[ \t\n] {} 
. {return(UNKNOWN);}
--.* {return(COMMENT);}
%%

For each of the following inputs, write the token stream that will be generated by the above Lex specification.

<table>
<thead>
<tr>
<th>Input</th>
<th>Token Stream</th>
</tr>
</thead>
<tbody>
<tr>
<td>class class1</td>
<td></td>
</tr>
<tr>
<td>$-123.0e+6</td>
<td></td>
</tr>
<tr>
<td>class --class1</td>
<td></td>
</tr>
<tr>
<td>return return21</td>
<td></td>
</tr>
</tbody>
</table>
3. [Total: 10 points] Consider the context-free grammar $G_3$ with start symbol $S$:

\[
\begin{align*}
S & \rightarrow TU \\
T & \rightarrow aT \mid \epsilon \\
U & \rightarrow abU \mid \epsilon
\end{align*}
\]

a. (3 points) Compute $\text{FIRST}$ and $\text{FOLLOW}$ sets for the three nonterminal symbols, $S$, $T$ and $U$.

b. (2 points) Is $G_3$ $LL(1)$? Justify.
c. (2 points) Show that $G_3$ is not SLR(1).
d. (3 points) A small modification to $G_3$ will make it $SLR(1)$. Suggest such a modification and show that the resultant grammar is $SLR(1)$. 
4. [Total: 8 points] Consider the following simplified grammar, \( G_4 \), of a formatting language for math equations:

\[
\begin{align*}
T & \rightarrow T \exp T \\
T & \rightarrow \sqrt{T} \quad \text{(Exponent)} \\
T & \rightarrow T \frac{T}{T} \\
T & \rightarrow (T) \\
T & \rightarrow \text{id} \\
T & \rightarrow \text{TOK}\_exp \ T \\
T & \rightarrow T \text{TOK}\_exp T \quad \text{(rule 1)} \\
T & \rightarrow T \ . \ \text{TOK}\_frac T \quad \text{(rule 3)} \\
\text{TOK}\_exp & \text{ shift, and go to state 7} \\
\text{TOK}\_frac & \text{ shift, and go to state 8} \\
\text{TOK}\_exp & \text{ [reduce using rule 1 (t)]} \\
\text{TOK}\_frac & \text{ [reduce using rule 1 (t)]} \\
\$default & \text{ reduce using rule 1 (t)}
\end{align*}
\]

a. (2 points) When grammar \( G_4 \) is specified in bison/yacc, we get 6 shift/reduce conflicts. In the .output file produced by bison/yacc, one of the states with conflicts is reported as:

state 10

\[
\begin{align*}
T & \rightarrow T \ . \ \text{TOK}\_exp T \quad \text{(rule 1)} \\
T & \rightarrow T \text{TOK}\_exp T \ . \quad \text{(rule 1)} \\
T & \rightarrow T \ . \ \text{TOK}\_frac T \quad \text{(rule 3)} \\
\text{TOK}\_exp & \text{ shift, and go to state 7} \\
\text{TOK}\_frac & \text{ shift, and go to state 8} \\
\text{TOK}\_exp & \text{ [reduce using rule 1 (t)]} \\
\text{TOK}\_frac & \text{ [reduce using rule 1 (t)]} \\
\$default & \text{ reduce using rule 1 (t)}
\end{align*}
\]

What entries in the parsing table (i.e., what state and terminal symbols) do the above conflicts correspond to?
b. (3 points) The shift/reduce conflicts can be eliminated by specifying the precedence and associativity information for the operators, i.e., exp, sqrt and frac, in G₄. The following examples (strings in the language specified by G₄, and the corresponding math formulas) illustrate the associativity and precedence of the three operators:

<table>
<thead>
<tr>
<th>String</th>
<th>Formula</th>
</tr>
</thead>
<tbody>
<tr>
<td>a exp b frac n</td>
<td>( \frac{a^b}{n} )</td>
</tr>
<tr>
<td>a exp n exp b</td>
<td>a^{nb}</td>
</tr>
<tr>
<td>sqrt a frac n</td>
<td>( \frac{\sqrt{n}}{a} )</td>
</tr>
<tr>
<td>sqrt a exp n</td>
<td>( \sqrt{a^n} )</td>
</tr>
<tr>
<td>a frac b frac c</td>
<td>( \frac{a}{bc} ) equivalent to ( \frac{a}{b^c} )</td>
</tr>
</tbody>
</table>

Specify, in the form of yacc/bison declarations, the operator precedence and associativity information for G₄ that is consistent with the above examples.

c. (3 points) How will the shift/reduce conflicts in state 10 be resolved by bison/yacc if the precedences and associativity you defined in part (b) were added to the bison/yacc specifications?