Please read all instructions (including these) carefully.

There are 5 questions on the exam, all with multiple parts. You have 75 minutes to work on the exam.

The exam is closed book, closed notes, closed computers, phones, tablets, etc.

Please write your answers in the space provided on the exam, and clearly mark your solutions. Please do not use any additional scratch paper.

Solutions will be graded on correctness and clarity. Each problem has a relatively simple and straightforward solution. You may get as few as 0 points for a question if your solution is far more complicated than necessary. Partial solutions will be graded for partial credit.

NAME: ____________________________________________

UT email address: _________________________________

EID: _________________________________

In accordance with the letter and the spirit of the UT Austin honor code I have neither given nor received assistance on this examination.

SIGNATURE: ____________________________________

<table>
<thead>
<tr>
<th>Problem</th>
<th>Max points</th>
<th>Points</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>20</td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>15</td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>20</td>
<td></td>
</tr>
<tr>
<td>4</td>
<td>15</td>
<td></td>
</tr>
<tr>
<td>5</td>
<td>10</td>
<td></td>
</tr>
<tr>
<td>Total</td>
<td>80</td>
<td></td>
</tr>
</tbody>
</table>
1. (20 points) Lambda Calculus

(a) (5 points) Reduce the following lambda calculus expression as much as possible. If the expression does not converge, write “divergent”.

$$(\lambda x.x x)(\lambda x.x x)$$

Solution: $$(\lambda x.x x)(\lambda x.x x) \rightarrow^\beta (x x)[(\lambda x.x x)\backslash x] \rightarrow^\beta (\lambda x.x x)(\lambda x.x x) \rightarrow^\beta \text{divergent}$$

(b) (5 points) Reduce the following lambda calculus expression as much as possible. If the expression does not converge, write “divergent”.

$$(\lambda b.b)((\lambda a.a)(\lambda b.a b))$$

Solution: $$(\lambda b.b)((\lambda a.a)(\lambda b.a b)) \rightarrow^\beta (\lambda b.b)(\lambda b.a b) \rightarrow^\beta (\lambda b.a b)$$
(c) (10 points) Is the following expression a fixed point operator in Lambda calculus?

\[ Z = \lambda f. (\lambda x.x)(\lambda x.f(xx)) \]

Prove or disprove your answer.

Solution: Yes, this is a fixed point operator. To prove this, we need to show that 
\[ Z h = h(Z h) \] for any function \( h \).

\[ (\lambda f. (\lambda x.x)(\lambda x.f(xx)))h \rightarrow (\lambda x.x)(\lambda x.h(xx)) \rightarrow (\lambda x.h(xx))(\lambda x.h(xx)) \rightarrow h((\lambda x.h(xx))(\lambda x.h(xx))) \]

Here, the last two expressions show that 
\[ Z h = h(Z h) \].
2. (15 points) Consider the following string over the alphabet $\Sigma = \{a, b, c\}$:

$abaaabbbcbbaa$

Give a flex specification that tokenizes this string as follows containing no more than two rules with no more than 4 symbols each:

$ab|a|a|abbb|cb|a|a$

Solution:

$ab^* \{ \} \$

$cb \{ \}$


3. (20 points) Regular Languages

(a) (7 points) Write a regular expression over the alphabet \( \Sigma = \{a, b, c\} \) for all strings such that every \( a \) is followed by at least one \( c \) before the next \( a \). Do not use flex notation.

Solution: \( ((ab^*c^+) + b)^* \)

(b) (7 points) Draw an NFA for the language from (a). Your NFA must have no more than four states.

Solution:

\[ 
\begin{array}{c}
  b, c \\
  \downarrow \\
  a \\
  \downarrow \\
  c \\
  \downarrow \\
  b \\
\end{array}
\]
(c) (6 points) Give a context-free grammar that generates the same language.

Solution:

\[
\begin{align*}
S & \rightarrow \varepsilon \mid bS \mid cS \mid aS'cS \\
S' & \rightarrow \varepsilon \mid S'b
\end{align*}
\]
4. (15 points) Consider the following context free grammar describing strings in the alphabet \( \Sigma = \{a, b, c\} \):

\[
S \rightarrow \varepsilon \mid bS \mid cS \mid aScS
\]

(a) (5 points) Give a one-sentence (no more that 20 words) description of the language described by this grammar.

Solution: Every \( a \) is followed by at least one \( c \).

(b) (10 points) Is this language regular? If so, give a regular expression that generates the same language. If not, explain clearly why not.

Solution: This is not a regular language. It is possible to have an unbounded number of \( a \)'s before any \( c \), and to ensure that every \( a \) is matched by at least one \( c \), we need to count this unbounded number of \( a \)'s. This cannot be done by a DFA.
5. (10 points) Short answer questions:

(a) (5 points) Is it possible to write a context-free grammar for every regular expression? Explain your answer.

Solution: Of course. \( L(\text{Regular}) \subseteq L(\text{CFG}) \)

(b) (5 points) We saw in lecture that any DFA can be converted to an NFA that accepts the same language. Is it also possible to convert each NFA to a DFA? If so, explain how, if not, justify why not.

Solution: Every DFA is also an NFA.