COMP603: MIDTERM I

NAME: ____

Complete within 120 minutes. Read each question carefully. Write legibly and check your work. No calculators, phones, or laptops are allowed. Good luck!

1. Short Definitions

Correctly define 8 of the following terms for full credit. Correctly define all for extra credit.

(1) String `sequence of characters/symbols.`

(2) Language `set of strings.`

(3) Compiler `translates a source language to a target language.`

(4) Interpreter `executes source code.`

(5) Bootstrapping `the process of writing a compiler in its own language by first writing a compiler for the language in an existing language.`

(6) Visitor `design pattern for tree traversal.`

(7) Nondeterminism `having a choice about which state to transition to.`

(8) Ambiguity `more than one parse of a string is possible.`

(9) First set `the set of terminals appearing first in any string derived from a nonterminal.`

(10) Follow set `the set of terminals appearing first in any string after deriving a nonterminal.`

2. Lists

Complete 3 of the following lists for full credit. Complete all for extra credit.

(1) Compiler phases, in order. Briefly describe what each phase does.

(a) `Scanner/Lexer/Tokenizer. Converts a string into a token sequence.`

(b) `Parser. Constructs a parse tree.`

(c) `Semantic analysis. Type checking.`

(d) `Optimizer. Improve performance.`

(e) `Code generator. Output code.`

(2) Primitive regular expressions. Briefly describe what each regular expression matches.

(a) `Alternation. a|b a or b.`

(b) `Catenation. ab a followed by b.`

(c) `Kleene Closure a* a 0 or more times.`

(d) `Empty Set. {} Nothing.`

(e) `Empty String. ""`

(f) `Symbol. A character.`

(3) Finite automaton elements. Describe each.

(a) `Set of states.`

(b) `Start state. The initial state.`

(c) `Set of transitions. State x Symbol -> State`

(d) `Set of accepting states. (Final states)`

(e) `Alphabet. The set of symbols.`

(4) For a grammar to be LL(1),[1] it must be:

(a) `Unambiguous, recognized by a deterministic pushdown automaton`

(b) `Free of left recursion`

(c) `Free of common prefixes`

(d) `First(A) disjoint from Follow (A)`

3. Fill in the blank

Complete the following statements for full credit.

(1) A pushdown automaton is a finite automaton with `one stack.`

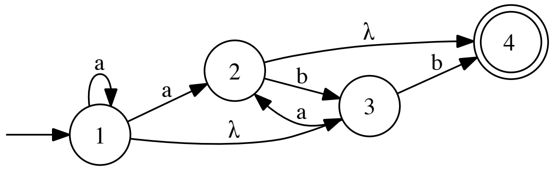
(2) A Turing machine is a finite automation with `an infinite read/write tape.`

(3) It is `never` possible to define an NFA which cannot be converted into a DFA.

[1] Left-right, Leftmost derivation, 1 token lookahead

4. Regular languages

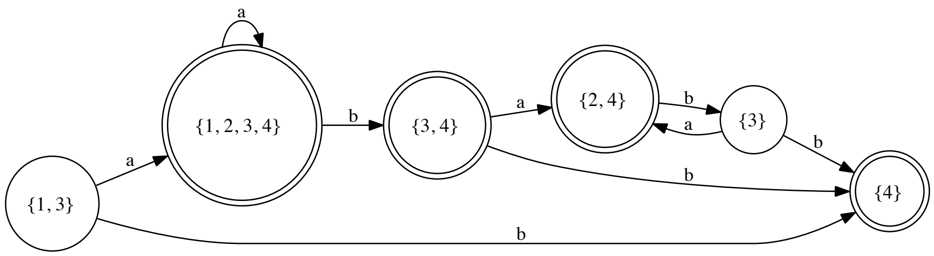
Refer to the Figure below. Answer 3 of the following questions. Answer all for extra credit.



(1) What is the initial state of the DFA using subset construction?

`{1,3}`

(2) Draw the equivalent DFA using subset construction.



(3) Write the equivalent regular expression.

$b|aa*(\lambda|b|bb|ba(ba)*(b|\lambda))$

(4) IPv4 addresses are written as four integers, separated by dots (e.g., `173.203.204.223`). Each integer ranges from 0 to 255. Write a regular expression to match precisely these addresses.

$(25[0-5]|2[0-4][0-9]|1[0-9]\{1,2\}|[0-9]\backslash.)\{3\}25[0-5]|2[0-4][0-9]|1[0-9]\{1,2\}|[0-9]$

5. Context-free languages

Refer to the context-free grammar below. S is the start symbol. Answer 4 of the following questions. Answer all for extra credit.

$$
\begin{array}{ll}
S \to T & T \to \mathbf{x} \\
S \to S + T & T \to \mathbf{y} \\
S \to S - T & T \to \mathbf{z} \\
S \to S * T & T \to (S) \\
S \to S/T &
\end{array}
$$

(1) Is the grammar above ambiguous? Why or why not?

```
Unambiguous, because the ( symbol forces the parse tree to be formed one way.
```

(2) Explain why the grammar above is not LL(1).

```
Left recursion, common prefixes.
```

(3) What is $First(T)$?

```
x, y, z, (
```

(4) What is $Follow(S)$?

```
+, -, *, /, ), eof
```

(5) Perform a leftmost derivation of the following string: $\mathbf{x} * (\mathbf{y} + \mathbf{z})$

```
S => S * T
  => T * T
  => x * T
  => x * (S)
  => x * (S + T)
  => x * (T + T)
  => x * (y + T)
  => x * (y + z)
```

6. Extra credit

Complete any of the following for extra credit.

(1) List all possible *Sentences* that can be matched by the grammar below.

$$
\begin{aligned}
&\mathit{Sentence} \to \mathit{NounPhrase}\ \mathit{VerbPhrase} && \mathit{Noun} \to \textbf{boy}\\
&\mathit{NounPhrase} \to \mathit{Article}\ \mathit{Noun} && \mathit{Noun} \to \textbf{ball}\\
&\mathit{VerbPhrase} \to \mathit{Verb}\ \mathit{NounPhrase} && \mathit{Verb} \to \textbf{kicked}\\
&\mathit{Article} \to \textbf{the} &&
\end{aligned}
$$

```
the boy kicked the ball
the boy kicked the boy
the ball kicked the boy
the ball kicked the ball
```

(2) Rewrite the grammar on the previous page to be LL(1).

```
Must eliminate left recursion, common prefixes.

S -> T Y
T -> x
  -> y
  -> z
  -> (S)
Y -> + T
  -> - T
  -> * T
  -> / T
  -> lambda
```