Module-2 Practice Questions

Specification and recognition of Tokens

1. Write the regular definitions (specifications) for the following tokens
keywords (if, then, else), identifier, relop, num, white space (ws)
Illustrate the recognition of above token with transition diagrams.

2. Illustrate the simple approach to the design of a lexical analyzer for an identifier.

LEX: Lexical Analyzer Generator

3. A Lex program is given below –

Regular Definitions
None
Transition Rules
a { } /* actions are omitted here */
abb { }
a^*b^+ { }

Implement the Lex as DFA.

4. A Lex program is given below –

Auxiliary Definitions
None
Transition Rules
a { } /* actions are omitted here */
abb { }
a^*b^* { }

Implement the Lex as DFA.

5. Write a Lex program to recognize the following tokens –
ws {white space}, if, then, else, end, id
Also generate the output of Lex as Lexical Analyzer (Finite Automata).

CFG: Derivations, Derivation tree, ambiguity in a grammar

6. **(a)** Derive left and right most derivations for the input string ***a=b⋆c+d/e*** for the given grammar
E→E+E|E-E|E⋆E|E/E|(E)|id

(b) Find the left most and right most derivations for the string **+⋆-xyx**in the grammar:
E→+EE|⋆EE|-EE|x|y

(c) Consider G whose productions are **S →** ***aAS*** **I** ***a, A*** **→** ***SbA*** **| SS |** ***ba***. Show that
S => *aabbaa* and construct a derivation tree whose yield is *aabbaa.*

(c) Let G be the grammar **S →** ***0B*** **|** ***1A, A*** **→ 0 | 0S |** ***1AA, B*** **→ 1 | 1S |** ***0BB***.
For the string 00110101, find **(i)** the leftmost derivation, **(ii)** the rightmost derivation, and **(iii)** the derivation tree.

(c) If G is the grammar **S** $\rightarrow$ ***SbS*** **|** ***a***, show that G is ambiguous.
(d) If G is the grammar **S** $\rightarrow$ ***S+S*** **|** ***S*S*** **|** ***a*** **|** ***b***, show that G is ambiguous.
(e) Show that the grammar **S** $\rightarrow$ ***a*** **|** ***abSb*** **|** ***aAb***, ***A*** $\rightarrow$ ***bS*** **|** ***aAAb*** is ambiguous.
(f) Show that the grammar $S \rightarrow aB \mid ab,\ A \rightarrow aAB \mid a.\ B \rightarrow ABb \mid b$ is ambiguous.

CFG: Simplification of CFGs

7. **(a)** Find a reduced grammar equivalent to the grammar $S \rightarrow aAa,\ A \rightarrow bBB,\ B \rightarrow ab,\ C \rightarrow aB$.
(b) Given the grammar $S \rightarrow AB,\ A \rightarrow a,\ B \rightarrow C \mid b,\ C \rightarrow D,\ D \rightarrow E, E \rightarrow a,$ find an equivalent grammar which is reduced and has no unit productions.
(c) Construct a reduced grammar equivalent to the grammar
$S \rightarrow aAa,\ A \rightarrow Sb \mid bCC \mid DaA.\ C \rightarrow abb \mid DD,\ E \rightarrow aC,\ D \rightarrow aDA$
(d) Consider the grammar G whose productions are $S \rightarrow as \mid AB,\ A \rightarrow \Lambda,\ B \rightarrow \Lambda,\ D \rightarrow b.$ Construct a grammar G_1 without null productions generating $L(G) - \{ \Lambda \}$.

PDA: Construction of a PDA for a given Context-free Language (CFL)

8. **(a)** Define Push Down Automata. Construct Push Down Automata accepting the following language $\mathbf{L = \{a^n b^n | n \geq o\}}$
(b) Construct the PDA that recognizes the languages $L=\{x=x^R: x \in \{a,b\}^+\}$.
(c) Design Push Down Automata for $L = \{a^n b^n | n \geq 1\}$.
(d) Design Push Down Automata for $L = \{a^{2n} b^n \mid n \geq 1\}$.
(e) Design Push Down Automata for $L = \{a^n b^{2n} \mid n > 0\}$.
(f) Design PDA for $L = \{a^n b^n c^m \mid n, m \geq 1\}$.
(g) Design PDA for $L = \{a^m b^n c^n \mid n, m \geq 1\}$.
(h) Design PDA for $L = \{a^n b^m c^n \mid n, m \geq 1\}$.
(i) Construct the PDA that recognizes the languages $L=\{\mathbf{ww^R} | w \in (0+1)^*,\ \mathbf{w^R}$ is reverse of $\mathbf{w}\}$.

Note: Acceptance might be asked by either final state or by null store (empty stack).

PDA: Conversion of CFG to PDA

9. **(a)** Construct the PDA to the following grammar:
$S \rightarrow aAA,\ A \rightarrow aS \mid bS \mid a$
(b) Convert the following Context Free Grammar to Push Down Automata
$S \rightarrow aSbb \mid aab$
(c) Find Push Down Automata that accepts the Context-free Grammar:
$S \rightarrow XY, \quad X \rightarrow aX \mid bX \mid a, \quad Y \rightarrow Ya \mid Yb \mid a$
(d) Construct PDA for the given CFG: $\mathbf{S \rightarrow aSb, S \rightarrow ab}$, Where **S** is the only variable and **{a,b}** are terminals.
(e) Convert the following grammar in a PDA that accepts the language by empty stack
$S \rightarrow 0S1 \mid A, \quad A \rightarrow 1A0 \mid S \mid \varepsilon$

PDA: Conversion of PDA to CFG

10. (a) Convert the following Push Down Automata to Context Free Grammar

$M = (\{q_0, q_1\}, \{a,b\} \{z_0, za\}, \delta, q_0, z_0, \varphi)$ where δ is given by

$\delta(q_0, a, z_0) = (q_0, z_a z_0)$

$\delta(q_0, a, z_a) = (q_0, z_a z_a)$

$\delta(q_0, b, z_a) = (q_1, \epsilon)$

$\delta(q_1, b, z_a) = (q_1, \epsilon)$

$\delta(q_1, \epsilon, z_0) = (q_1, \epsilon)$

(b) Construct CFG for the PDA given below $A=(\{q_0,q_1\},\{0,1\},\{S,A\},\delta, q_0,S,\varphi)$ where δ is given as below

$\delta(q_0,1,S)=\{(q_0,AS)\}$

$\delta(q_0,\varepsilon,S)=\{(q_0,\varepsilon)\}$

$\delta(q_0,1,A)=\{(q_0,AA)\}$

$\delta(q_0,0,A)=\{(q_1,A)\}$

$\delta(q_0,1,A)=\{(q_1,\varepsilon)\}$

$\delta(q_1,0,S)=\{(q_0,S)\}$

(c) Construct CFG for the PDA $M=(\{q_0,q_1\},\{0,1\},\{R,Z_0\},\delta, q_0,Z_0,\varphi)$ where δ is given by

$\delta(q_0,1,Z_0)=\{(q_0,RZ_0)\}$

$\delta(q_0,1,R)=\{(q_0,RR)\}$

$\delta(q_0,0,R)=\{(q_1,R)\}$

$\delta(q_1,0,Z_0)=\{(q_0,Z_0)\}$

$\delta(q_0,\varepsilon,Z_0)=\{(q_0,\varepsilon)\}$

$\delta(q_1,1,R)=\{(q_1,\varepsilon)\}$

<u>Proving the given language L as not context-free using Pumping lemma</u>

11. Using pumping lemma Prove that the following languages are not context-free.

(a) $L = \{a^n b^n c^n \mid n \geq 1\}$

(b) $L = \{a^p \mid p \text{ is prime}\}$

(c) The set of all strings over $\{a, b, c\}$ in which the number of occurrences of a, b, c is the same.

(d) $L = \{a^m b^m c^n \mid m \leq n \leq 2m\}$

(e) $L = \{a^m b^n \mid n = m^2\}$