Computer Science and Engineering Department, IIT Kharagpur

CS31003 Compilers, 3rd year CSE, 5th Semester **(Class Test 1)**

Time limit: 1 hour Date: 30th Aug, 2024 Max Marks: 20

Roll No: ______________ **Name:** ____

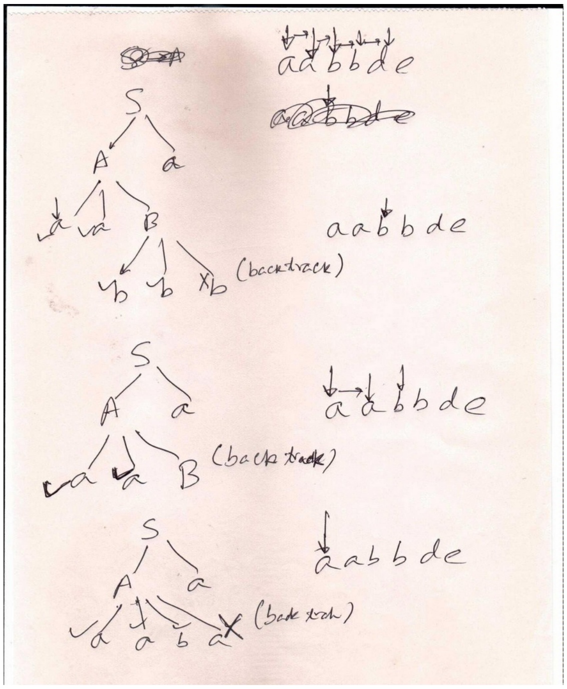
Answer all the questions. Take and state suitable assumptions, if needed.
No clarifications will be provided during the examination.

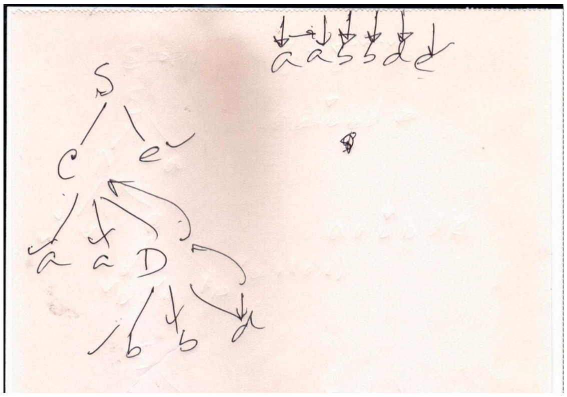
1. Parse the string ***aabbde*** by executing the non-deterministic recursive descent parser under the grammar specified below. Apply the productions in the increasing order of the Rule #. Clearly show the functions invoked by the parser and the backtracking steps (if any). No need to write the codes for the functions.

Rule #	**Production**
1.	$S \rightarrow Aa$
2.	$S \rightarrow Ce$
3.	$A \rightarrow aaB$
4.	$A \rightarrow aaba$
5.	$B \rightarrow bbb$
6.	$C \rightarrow aaD$
7.	$D \rightarrow bbd$

[5]

Ans:





2. Consider the following grammar with terminal symbols $\{a, b, d, g, h\}$, and nonterminals $\{S, A, B, C\}$. Here, S is the start symbol. The productions of the grammar are given below. The bodies of two productions, called **Body1** and **Body2**, are missing.

$S \rightarrow$ **Body1** $| \ CbB \ | \ Ba$
$A \rightarrow da \ |$ **Body2**
$B \rightarrow g \ | \ \epsilon$
$C \rightarrow \ h \ | \ \epsilon$

The following table lists the FIRST and FOLLOW of the nonterminals.

	FIRST	FOLLOW
S	$\{d, g, h, b, a, \epsilon\}$	$\{\$\}$
A	$\{d, g, h, \epsilon\}$	$\{h, g, \$\}$
B	$\{g, \epsilon\}$	FOLLOW(S) ∪ $\{a\}$ ∪ FIRST(C) ∪ FOLLOW(A)
C	$\{h, \epsilon\}$	FIRST(B) ∪ FOLLOW(S) ∪ FOLLOW(A) ∪ $\{b\}$

Using this table, derive the missing bodies **Body1** and **Body2** of the two productions given above. Show all the steps of your derivation.

[5]

Ans:

$A \rightarrow BC$ [First of A includes g and h, where First of B is g and First of C is g*]*

$S \rightarrow ABC$ [First of S includes g, h, which infers B, C in production body. First of S includes d, which infers A in production body. Follow of B includes First of C, Follow of A is $\{d, g, h, \epsilon\}$, and does not include Follow(S). This decides the order ABC.]

3. (a) Consider a programming language, which supports the following tokens.

ID: Identifier without numeric digits
ID_N: Identifier with numeric digits
INT: Signed integers without decimal point
NE_REAL: Signed real numbers with decimal points but without exponents
E_REAL : Signed real numbers with decimal points and exponents
Keywords: int, float
Punctuator: ;
OP: operators such as =
WS: White spaces

Write down the regular definitions of the tokens **INT**, **NE_REAL**, **E_REAL**, **ID**, **ID_N**.

[2]

Ans:

INT : digit(digit)*[.digit]?(digit)*[E[+/-]?digit]?digit*
NE_REAL : (epsilon/+/-)digit(digit)*[.digit]?(digit)*
E_REAL : (epsilon/+/-)digit(digit)*[.digit]?(digit)*[E[+/-]?digit]?digit*
ID: letter+
ID_N: letter(letter+digit)*

For the code snippet below, write down the stream of tokens generated by the lexical analyzer. Write each input token as <token_name, lexeme>.

```
int var = 250e-2;
float var2 = -20.45;
```

[1]

Ans:

<Keyword, int><ws><ID,var><=><E_REAL, 250e-2><Punctuator, ;>
<Keyword, float><ws><ID_N,var2><=><NE_REAL, -20.45><Punctuator, ;>

(b) Consider a programming language L which supports three tokens **T1**, **T2** and **T3** defined by the regular expressions **T1** = ***a*?(*b*|*c*)*a**, **T2** = ***b*?(*a*|*c*)*b**, **T3** = ***c*?(*b*|*a*)*c**. Consider a string ***w*** = ***accbbbccaabc*** in the language L. Arrange these tokens in such a way that the tokens generate the string ***w***, satisfying the following two conditions. (i) The number of tokens should be minimized, and (ii) the tokens cannot be repeated. [It is not necessary that you have to use all these three tokens.]

[3]

Ans:

T1 will generate *accbbbcca* and T3 will generate *abc*. So the sequence of tokens are T1T3.

4. (a) Consider the following grammar with terminal symbols $\{a, b, c, d\}$ and nonterminal symbols $\{X, Y\}$, where X is the start symbol. Eliminate left recursion from the grammar, and write down the transformed grammar.

$X \rightarrow Ya \mid Xa \mid c$
$Y \rightarrow Yb \mid Xb \mid d$

[2]

Ans:

$X \rightarrow Ya \mid Xa \mid c$ … Eliminate left recursion

$X \rightarrow YaX' \mid cX'$
$X' \rightarrow aX' \mid \epsilon$

$Y \rightarrow Yb \mid Xb \mid d$ … Substitute with X productions here
$Y \rightarrow Yb \mid YaX'b \mid cX'b \mid d$ … Eliminate left recursion now

$Y \rightarrow cX'bY' \mid dY'$
$Y' \rightarrow bY' \mid aX'bY' \mid \epsilon$

(ii) Final productions:

$X \rightarrow YaX' \mid cX'$
$X' \rightarrow aX' \mid \epsilon$
$Y \rightarrow cX'bY' \mid dY'$
$Y' \rightarrow bY' \mid aX'bY' \mid \epsilon$

(b) Prove or disprove with justification: The following grammar is LL(1). Do not construct the parsing table. Here, S is the only nonterminal symbol.

$S \rightarrow aSbS \mid bSaS \mid \epsilon$

[2]

Ans:

Follow(S)=$\{a, b, \$\}$
First($aSbS$)=$\{a\}$ overlaps with Follow(S)