1. Consider the grammar given below:

S → NP VP	Det → *that* | *this* | *a* | *the*
S → Aux NP VP	Noun → *book* | *flight* | *meal* | *man*
S → VP	Verb → *book* | *include* | *read*
NP → Det NOM	Aux → *does*
NOM → Noun	
NOM → Noun NOM	
VP → Verb	
VP → Verb NP	

i) What can be told about the above grammar? (1 Mark)
   a) It has 3 unit productions
   b) It has 6 unit productions
   c) Certain terminals in the grammar cannot be derived
   d) Few useless symbols are inherently present in the grammar

ii) I: Regular grammars are a subset of Context Free Grammars (1 Mark)
   II: Context free grammars are accepted by FSA
   a) I is true and II is false
   b) Both I and II are true
   c) II is true and I is false
   d) Both are false

iii) List the terminals and non-terminals in the given grammar (3 Marks)

iv) Check if the above grammar could generate the string "The flight include meal and the man read a book" (5 Marks)

v) Simplify the grammar and then convert the above CFG to Chomsky Normal Form (CNF) (8+7 Marks)

(i) a) It has 3 unit productions

(ii) a) I is true and II is false

(iii) Terminals → { that, this, a, the, book, flight, meal, man, include, read, does }

Non-Terminals → { S, NP, VP, Aux, Det, Nom, Verb, Noun }

(iv) string : "the flight include meal and the man read a book"

S ⇒ NP VP

S ⇒ Det NOM VP

S ⇒ the NOM VP

S ⇒ the Noun VP

S ⇒ the flight VP

S ⇒ the flight Verb NP

S ⇒ the flight include NP

S ⇒ the flight include det NOM

S ⇒ the flight include ?

So, from det, there is no possible transition which can give "meal" as string and also "and" is not present in any of the transition which is required for string to derive. Hence above grammar cannot generate given string.

(V) Simplification of Grammar

Step1 : Eliminate ε production

There is no ε production.

step2 : Eliminate unit production

Unit Pairs	Productions
(S, S)	S → NP VP | Aux NP VP
(S, VP)	S → Verb NP
(S, verb)	S → book | include | read
(NP, NP)	NP → Det NOM
(NOM, NOM)	NOM → Noun NOM
(NOM, Noun)	NOM → book | flight | meal | man
(VP, VP)	VP → Verb NP

(VP, Verb)	VP → book | include | read
(Det, Det)	Det → that | this | a | the
(Noun, Noun)	Noun → book | flight | meal | man
(Verb, Verb)	Verb → book | include | read
(Aux, Aux)	Aux → does

Now the grammar is,

S → NP VP | Aux NP VP | Verb NP | book

S → include | read

NP → Det NOM

NOM → Noun Nom | book | flight | meal | man

VP → Verb NP | book | include | read

Det → that | this | a | the

Noun → book | flight | meal | man

Verb → book | include | read

Aux → does

Step3 : Eliminate useless symbols

All symbols are generating & reachable

Hence the final grammar is,

S → NP VP | Aux NP VP | Verb NP | book |

S → include | read

NP → Det NOM

NOM → Noun NOM | book | flight | meal | man
VP → Verb NP | book | include | read
Det → that | this | a | the
Noun → book | flight | meal | man
Verb → book | include | read
Aux → does.

Conversion of CFG to CNF

Step1 : Optmize the grammar
The grammar is in optimized form.
Step2 : Introduce Non-terminals.
Not required
step 3 : Break Productions

S → NP VP | Aux P1 | Verb NP | book
S → include | read
NP → Det NOM
NOM → Noun NOM | book | flight | meal | man
VP → Verb NP | book | include | read
Det → that | this | a | the
Noun → book | flight | meal | man

Verb → book | include | read

Aux → does

PI → NP VP

2. Consider the grammar given below which denotes Boolean expressions

Expr → Expr or Term | Term

Term → Term and Factor | Factor

Factor → not Factor | (Expr) | true | false

i) Which of the following lemma/ algorithm is not related to CFG? (1 Mark)

a) Substitution rule

b) Elimination of left recursion

c) Pumping lemma for regular languages

d) Elimination of useless symbols

ii) A PDA can behave like a FSM when the stack size is____ (1 Mark)

a) 1 b) 0 c)n d) infinite

iii) Write leftmost derivation, rightmost derivation and parse tree for the string "true and not false or (false and (not true))" (6 Marks)

iv) Is the grammar ambiguous? (2 Marks)

v) Convert the given grammar to a Pushdown Automata (9 Marks)

vi) Show any two reachable Instantaneous Descriptions (IDs) starting from the initial ID for the string (6 Marks)

(i) c) Pumping Lemma for regular languages

(ii) b) 0

(iii) string :- true and not false or (false and (not true))

Left most derivation

$Expr \overset{lm.}{\Rightarrow}$ Expr or Term

$\Rightarrow$ Term or Term

$\Rightarrow$ Term and Factor or Term

$\Rightarrow$ Factor and Factor or Term

$\Rightarrow$ true and Factor or Term

$\Rightarrow$ true and not Factor or Term

$\Rightarrow$ true and not false or Term

$\Rightarrow$ true and not false or Factor

$\Rightarrow$ true and not false or (Expr)

$\Rightarrow$ true and not false or (Term)

$\Rightarrow$ true and not false or
(Term and Factor)

$\Rightarrow$ true and not false or
(Factor and Factor)

$\Rightarrow$ true and not false or
(False and Factor)

$\Rightarrow$ true and not false or
(False and (Expr))

$\Rightarrow$ true and not False or
(False and (Term))

$\Rightarrow$ true and not false or (False
and (Factor))

$\Rightarrow$ true and not False or (False
and (not Factor))

$\Rightarrow$ true and not False or (False
and (not true))

Right most derivation

Expr $\overset{rm}{\Rightarrow}$ Expr or Term

$\Rightarrow$ Expr or Factor

$\Rightarrow$ Expr or (Expr)

$\Rightarrow$ Expr or (Term)

$\Rightarrow$ Expr or (Term and Factor)

$\Rightarrow$ Expr or (Term and (Expr))

$\Rightarrow$ Expr or (Term and (Term))

$\Rightarrow$ Expr or (Term and (Factor))

$\Rightarrow$ Expr or (Term and (not Factor))

$\Rightarrow$ Expr or (Term and (not true))

$\Rightarrow$ Expr or (Factor and (not true))

$\Rightarrow$ Expr or (false and (not true))

$\Rightarrow$ Term or (false and (not true))

$\Rightarrow$ Term and Factor (false and (not true))

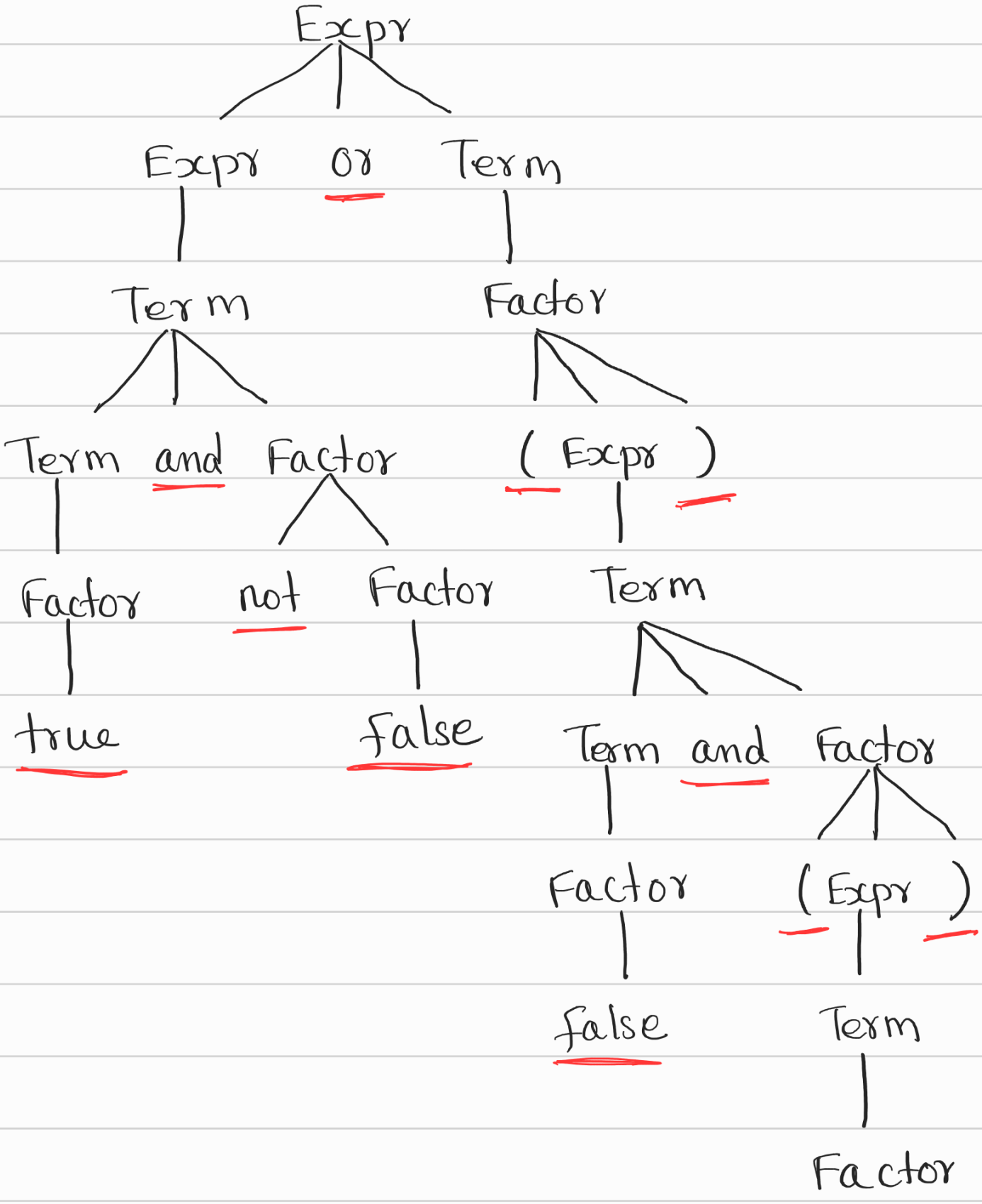
$\Rightarrow$ Term and not Factor (False and (not true))

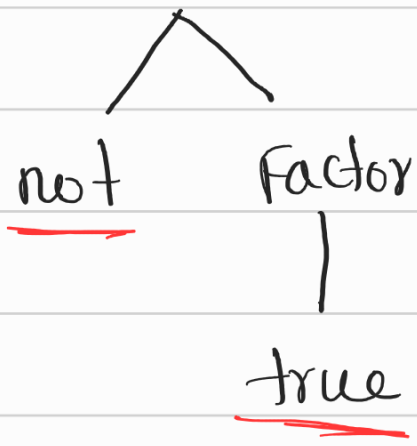
$\Rightarrow$ Term and not false (false and (not true))

$\Rightarrow$ Factor and not false (false

and (not true))

Parse Tree





(iv) Expr ⇒ Term
⇒ Term and Factor
⇒ true and Factor
⇒ true and not Factor

We can't find the different derivation for the given string. So the grammer is not ambiguous

(v) Convert grammar to PDA

Expr → Expr or Term | Term
Term → Term and Factor | Factor
Factor → not Factor | (Expr) | true | false

The grammar can be converted to a PDA that accepts by empty stack as follows:

$$P = (\{q\}, \Sigma, \Gamma, \delta, q_0, Z_0)$$

That is,

$$P = (\{q\}, \{or, and, not, (,), true, false\}, \{or, and, not, (,), true, false, Expr, Term, Factor\}, \delta, q, S)$$

where δ is defined as,

$\delta(q, \varepsilon, Expr) = \{(Expr \text{ or } Term), (Term)\}$

$\delta(q, \varepsilon, Term) = \{(Term \text{ and } Factor), (Factor)\}$

$\delta(q, \varepsilon, Factor) = \{(not\ Factor), ((Expr)), (true), (false)\}$

$\delta(q, or, or) = \{(q, \varepsilon)\}$

$\delta(q, and, and) = \{(q, \varepsilon)\}$

$\delta(q, not, not) = \{(q, \varepsilon)\}$

$\delta(q, (, () = \{(q, \varepsilon)\}$

$\delta(q,),)) = \{(q, \varepsilon)\}$

$\delta(q, true, true) = \{(q, \varepsilon)\}$

$\delta(q, false, false) = \{(q, \varepsilon)\}$

(vi) String(i):- true and not false (Any string can be taken)

Leftmost derivation (Not required)

$Expr \overset{lm}{\Rightarrow} Term$

$\Rightarrow$ Term and Factor

$\Rightarrow$ Factor and Factor

$\Rightarrow$ true and Factor

$\Rightarrow$ true and not Factor

$\Rightarrow$ true and not false

Simulation of the PDA

(q, true and not false, Expr)

$\vdash$ (q, true and not false, Term)

$\vdash$ (q, true and not false, Term and Factor)

$\vdash$ (q, true and not false, Factor and Factor)

$\vdash$ (q, true and not false, true and Factor)

$\nvdash$ (q, and not false, and Factor)

$\vdash$ (q, not false, Factor)

$\vdash$ (q, not false, not false)

$\vdash$ (q, false, false)

$\vdash$ (q, ε, ε)

String(ii): (false and (not true)) (Any string can be taken)

Left most Derivation (Not required)

Expr $\overset{lm}{\Longrightarrow}$ Term

$\Rightarrow$ Factor

$\Rightarrow$ (Expr)

$\Rightarrow$ (Term)

$\Rightarrow$ (Term and Factor)

$\Rightarrow$ (Factor and Factor)

$\Rightarrow$ (false and Factor)

$\Rightarrow$ (false and (expr))

$\Rightarrow$ (false and (term))

$\Rightarrow$ (false and (Factor))

$\Rightarrow$ (false and (not factor))

$\Rightarrow$ (false and (not true))

Simulation of the PDA

(q, (false and (not true)), Expr)

$\vdash$ (q, (false and (not true)), Term)

$\vdash$ (q, (false and (not true)), Factor)

$\vdash$ (q, (false and (not true)), (Expr))

$\vdash (q, \text{false and (not true)}, \text{Expr}))$

$\vdash (q, \text{false and (not true)}, \text{Term}))$

$\vdash (q, \text{false and (not true)}, \text{Term and Factor}))$

$\vdash (q, \text{false and (not true)}, \text{Factor and Factor}))$

$\vdash (q, \text{false and (not true)}, \text{false and Factor}))$

$\vdash (q, \text{and (not true)}, \text{and Factor}))$

$\vdash (q, \text{(not true)}, \text{Factor}))$

$\vdash (q, \text{(not true)}, \text{(Expr)}))$

$\vdash (q, \text{not true)}, \text{Expr)}))$

$\vdash (q, \text{not true)}, \text{Term)}))$

$\vdash (q, \text{not true)}, \text{Factor)}))$

$\vdash (q, \text{not true)}, \text{not Factor)}))$

$\vdash (q, \text{true)}, \text{Factor)}))$

$\vdash (q, \text{true)}, \text{true)}))$

$\vdash (q,),))$

$\vdash (q,),))$

$\vdash (q, \varepsilon, \varepsilon)$

3. A school organized an annual day celebration for all students. The students participated in various games of the events. One of the game is picking the ball from the bag. The student has to pick the balls in the order specified. The one who is picking all the balls in the specified order at the earliest is the winner. The coloured balls are Red, Green, Violet, and Yellow.

Case (i): First, they should pick 'n' number of red balls then 'm' number of green balls then 'm' number of Violet balls and at last 'n' number of yellow balls.

Case (ii): First they should pick 'n' number of red balls then 'm' number of green balls then 'm+n' number of Violet balls.

i. Which of the following is not considered as an application of CFG? (1 Mark)

   a. Natural Language processing
   b. Syntax checking in programming languages
   c. Speech recognition
   d. Mathematical Induction

ii. A symbol that cannot derive a terminal or any reachable symbol from the starting symbol is called as_____ (1 Mark)

   a. Null symbol
   b. Reachable symbol
   c. Non reachable symbol
   d. Useless symbol

iii. Construction of PDA for the case (i) with transition diagram (8 Marks)
iv. Construction of PDA for the case (ii) with transition diagram (12 Marks)
v. Instantaneous Description for picking 2 red balls, 1green ball and 2 violet balls from PDA in case (ii) (3 Marks)

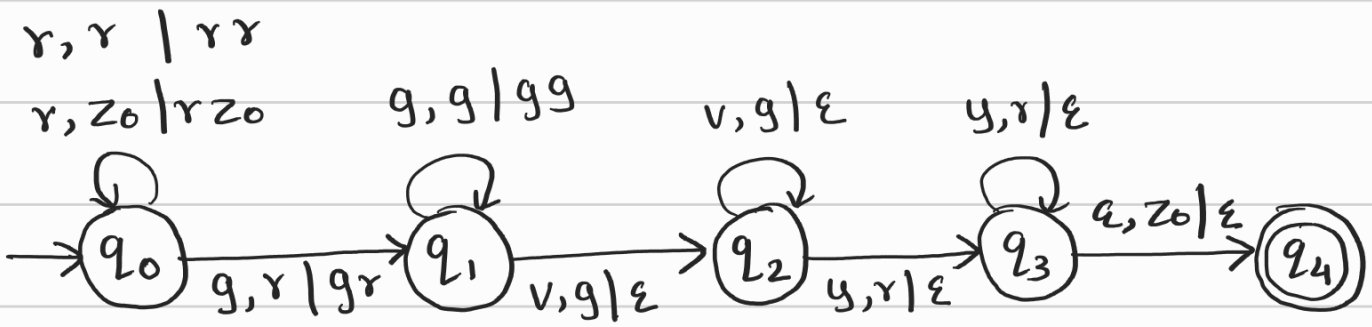
(i) d) Mathematical Induction

(ii) c) Non reachable Symbol

(iii) $L = \{ r^n g^m v^m y^n \}$

Logic for designing the PDA

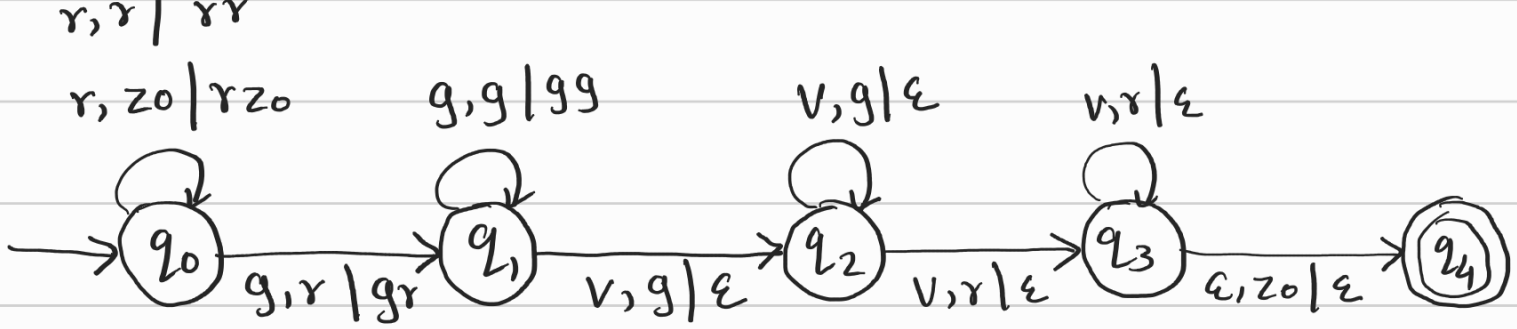
1. Push all r's into the stack.
2. Push all g's into the stack.
3. For every v we read, pop one g from the stack.
4. For every y we read, pop one r from the stack.
5. On consuming all the input if we reach the end of the stack, accept the input.



(iv) $L = \{ r^n g^m v^{m+n} \}$

Logic for designing the PDA

1. Push n r's into the stack.
2. Push m g's into the stack.
3. For m v's pop m g's and for n v's pop n r's from the stack.
4. When we finish reading the input, if the stack is empty, the input is accepted.



(v) $(q_0, rrgvv, z_0) \vdash (q_0, rgvv, rz_0)$

$\vdash (q_0, gvv, rrz_0)$

$\vdash (q_1, vv, grrz_0)$

$\vdash (q_2, v, rrz_0)$

$\vdash (q_2, \varepsilon, \boxed{r}z_0)$

The above string is not accepted because the stack not empty.