SRM INSTITUTE OF SCIENCE & TECHNOLOGY	**SRM INSTITUTE OF SCIENCE AND TECHNOLOGY** **DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING**

15CS301 THEORY OF COMPUTATION
IMPORTANT QUESTIONS
SIMPLE 4 MARK QUESTION WITH ANSWER

UNIT I

Part A

1. **Define the term NFA (Dec 2015)**

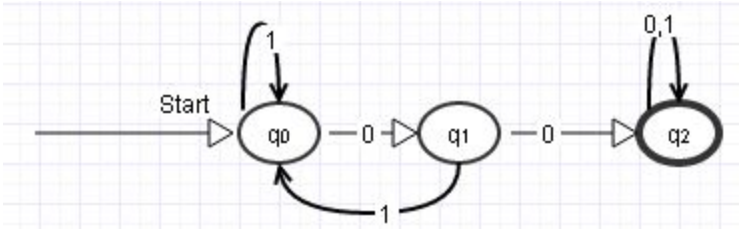
   A Non-deterministic Finite automaton is a mathematical model of computation used to design computer program. It is represented by a collection of 5 tuple (Q, Σ,δ,q_0,F) where,

   - Q is a finite set of states, which is non empty
   - Σ is input alphabet, indicates input set
   - q_0 is an initial state, q_0 is in Q
   - F is a set of final states
   - δ is a transition function

2. **Define deductive proof (Nov/Dec 2014)**

   Deductive proof consists of statement whose truth lead us from initial statement called hypothesis to a conclusion statement. If 'H' is hypothesis and 'C' is conclusion, then the statement would be *"if H then C"*

3. **Design DFA to accept strings over $\sum = (0,1)$ with two consecutive 0's (Nov/Dec 2014)**



4. **Prove or disprove that $(r+s)^* = r^* + s^*$ (Nov/Dec 2014)**

LHS	RHS
$(r+s)^*$	$r^* + s^*$
={ε, r,rr,s,ss,rs,sr,…}	={ε, r,rr,s,ss,rrr,sss,…}
= {ε, any combination of r and s}	= {ε, any combination of only r or any combination of only s}

5. **State the pumping lemma for regular languages. (Nov/Dec 2014,2013, June 2016)**

   In the theory of formal languages, the pumping lemma for regular languages describes an essential property of all regular languages. Informally, it says that all sufficiently long words in a regular language may be pumped (i.e), have a middle section of the word repeated an arbitrary number of times to produce a new word that also lies within the same language.

   Pumping Lemma : If A is a regular language, then there is a pumping length ***p*** such that:

   If ***s*** is any string in A of length at least p,

   Then ***s*** may be divided into three pieces, s= x**y**z, satisfying the following condition:

   - for each $i \geq 0$, $x\, y^i\, z \in A$
   - $|y| > 0$
   - $|xy| \leq p$

6. **What is a finite automaton (Nov/Dec 2014,Nov/Dec 2015)**

   Finite state automaton is a mathematical model of computation used to design computer program. Finite automata is represented by a collection of 5 tuple $(Q, \Sigma, \delta, q_0, F)$ where,

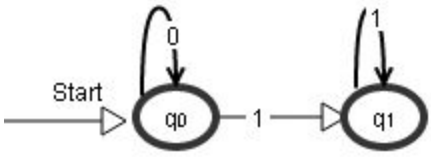
   - Q is a finite set of states, which is non empty
   - Σ is input alphabet, indicates input set
   - q_0 is an initial state, q_0 is in Q
   - F is a set of final states
   - δ is a transition function

7. **Enumerate the difference between DFA and NFA (Nov/Dec 2014)**

SN	DFA (May/Jun 2013)	NFA(Nov/Dec 2013)
1	Deterministic Finite state automaton is a mathematical model of computation used to design computer program. Finite automata is represented by a collection of 5 tuple $(Q, \Sigma, \delta, q_0, F)$ where, Q is a finite set of states, which is non empty Σ is input alphabet, indicates input set q_0 is an initial state, q_0 is in Q	Like DFA an NFA has finite set of input set, one start state and set of accepting states. The main difference between NFA and DFA is that in NFA, δ function takes a state and input symbol as argument as DFA, but it returns set of zero, one or more states. The NFA is represented by a collection of 5 tuple $(Q, \Sigma, \delta, q_0, F)$ where, Q is a finite set of states, which is non empty Σ is finite set of input alphabet,

	F is a set of final states δ is a transition function	q_0 is an initial state, q_0 is in Q F is a set of final states (accepting) state δ is a transition function
2	Every input string leads to the unique state FA	For the same input there can be more than one next states.
3	Conversion of regular expression (RE) of DFA is complex	Regular expression can be easily converted to NFA using Thompson's construction
4	The DFA requires more memory for storing the state information	The NFA requires more computations to match RE with input
5	The DFA it is not possible to move to next state without reading any symbol.	In NFA we can move to next state without reading any symbol.
6	$Q \times \sum = Q$	$Q \times \sum = 2^Q$

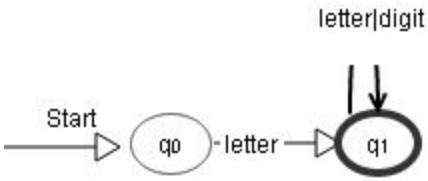
8. Construct a finite automaton for the regular expression 0^*1^* (May/Jun 2014)



9. Mention the closure properties of regular languages. (May/Jun 2014)

If ε-NFA's recognize the languages that are obtained by applying an operation on the ε -NFA recognizable languages then ε-NFA's are said to be closed under the operation. The ε-NFA's are closed under the following operations: Union, Intersection, Concatenation, Negation, Star, and Kleene closure

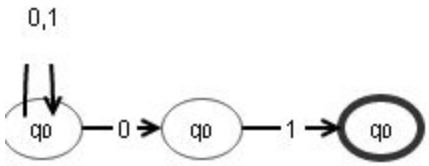
10. Draw the transition diagram (automata) for an identifier (Nov/Dec 2013)



11. (i)Construct a r.e for the language which accepts all strings with atleast two c's over the set Σ={c,b} (ii)Construct a r.e for the language over the set Σ={a,b} in which total number of a's are divisible by 3

Ans. (i) (b+c)* c (b+c)* c (b+c)* (ii) (b* a b* a b* a b*)*

12. Construct NFA equivalent to the regular expression (0+1)01 (Nov/Dec 2013)



13. **Differentiate L* and L^+**
    - L* denotes Kleene closure . If L is a set of symbols or characters then L* is the set of all strings over symbols in L, including the empty string ε Eg. 1* = {ε,1,11,111,1111,……}
    - L^+ denotes the positive closure (Kleen pluse). It is the set of all symbols / character in the set L except the empty string Eg. 1^+ = {1,11,111,……}
14. **Define the term Epsilon transition (May/Jun 2013, Dec 2015)**

    In the NFA the transition that does not require input symbols for state transition and is capable of transiting to zero or more states with ε is called epsilon transition.
15. **What is a regular expression (May/Jun 2013)**

    It is a sequence of character that defines a search pattern used for pattern matching. Let Σ be an alphabet used to denote the input set. the regular expression (RE) over Σ can be defined as
    - φ is a RE which denote the empty set
    - ∈ is a RE which denote the null string
    - For each 'a' in Σ is denoted as the set {a}
    - If 'r' and 's' are RE denoting the language L1 and L2 then
        - ' r + s' is equivalent to L1 ∪ L2
        - 'r s' is equivalent to L1 L2
        - 'r*' is equivalent to L1* (closure)
16. **Name any four closure properties of Regular languages. (May/Jun 2013)**
    a. The union of two regular language is regular
    b. The intersection of two regular languages is regular
    c. The complement of regular language is regular
    d. The closure operation on a regular language is regular
17. **State the principle of induction (Nov/Dec 2012)**

    The proof by Mathematical induction can be carried out using following steps:
    1. Basis: In this step we assume the lowest possible value.
    2. Induction Hypothesis: In this step we assign value of n to some other value K.
    3. Inductive Step: In this step, if n = k is true then we check whether the result is true for n = k + 1 or not. If we get the same result at n = k+1 then we can state that given proof is true by principle of mathematical induction.

18. **Give English description of the following language (0 + 10)*1***

The set of strings of 0's and 1's without any pair of consecutive 1's substring except at the end.

19. **What is Structural induction (Nov/Dec 2011)**

Structural induction is a kind of proof by induction. It is used to prove that there exists a relationship between some function of integers and a given formula. For this proof technique:

- Prove the pattern is true for smallest number
- Assume it holds for an arbitrary number n
- Prove that if it is true for n, then it must be true for n+1 as well

20. **What are the application of automata theory**

- In compiler construction.
- In switching theory and design of digital circuits.
- To verify the correctness of a program.
- Design and analysis of complex software and hardware systems.
- To design finite state machines such as Moore and mealy machines

21. **Give regular expression for the following: (Nov/Dec 2012)**
    a. **L1 = Set of all strings of 0 and 1 ending in 00**
    b. **L2 = set of all strings of 0 and 1 beginning with 0 and ending with 1**

Ans: L1 = (0+1)*00 L2 = 0 (0+1)* 1

Part B/Part C

THEORETICAL QUESTION

1. **State the Thomson construction algorithm and subset construction algorithm, Construct finite automata for generating any floating point number with an exponential factor for example numeric value of the form 1.23 e^{-10}. Trace for a string. (16) (June 2016)**
2. **Use mathematical induction to solve the problem of Fibonacci series and examine the relationship between recursive definition and proofs by induction. Also state the inductive proofs. (16) (June 2016)**
3. **Prove that "A language L is accepted by some DFA if and only if L is accepted by some NFA" (10) (Dec 2015)**
4. **Discuss on the relation between DFA and Minimal DFA (6) (Dec 2015)**

5. **Discuss on Finite automata with epsion transitions (6) (Dec 2015)**
6. **Let L be a set accepted by a NFA and then prove that there exists a DFA that accept L. (8) (Nov/Dec 2014)**
7. **Prove that a language L is accepted by some NFA if and only if L is accepted by some DFA (8) (Nov/Dec 2014)**
8. **State the pumping lemma for Regular language. Show that the set L =$\{0^{i2}|i \geq 1\}$ not regular (6) (Dec 2015)**
9. **Determine whether the following languages are regular or not with proper justification**
   a. **L1 = $\{a^n bc^{3n} \mid n \geq 0\}$ (8) (June 2016)**
   b. **L2 = $\{a^{5n} \mid n \geq 0\}$**
10. **Prove that $L = \{0^{i^2}$ / i is an integer i ≥ 1} is not regular (8) (May/June 2014)**
11. **Show that the given Language is not regular L = $\{a^n b^m \mid n<m$ and n,m≥1} (8 marks)**

PROBLEMATIC QUESTION

1. **Construct a NFA that accepts all strings the end in 01. Give its transition table and the extended transition function for the input string 00101. Also construct a DFA for the above NFA using subset construction method (10) (June 2016)**
2. **Prove the following by principle of induction** $\sum_{x=1}^{n} x^2 = \frac{n(n+1)(2n+1)}{6}$ **(June 2016) (6)**
3. **What is regular expression ? Write expression for set of strings that consists of alternating 0's and 1's (8) (June 2016)**
4. **Design a minimized DFA by converting the following regular expression to NFA, NFA-λ and to DFA over the alphabet Σ = {a,b,c}*. RE = a(a+b+c)* (a+b+c) (16) (June 2016)**
5. **Construct determine finite automata that recognize the regular expression defined over the alphabet Σ ={0,1}. RE = (1 +110)*0. Trace for a string acceptance and rejection. (8) (June 2016)**
6. **Construct Finite automata equivalent to the regular expression (ab+a)* (6) (Dec 2015)**
7. **Construct DFA to accept the language L = {w | w is of even length and begins with 10} (6) (Dec 2015)**
8. **Convert the following NFA to a DFA (10) (Dec 2015)**

	0	1

p	**{p,q}**	**{p}**
q	**{r,s}**	**{t}**
r	**{p,r}**	**{t}**
***s**	**φ**	**φ**
***t**	**φ**	**φ**

9. Construct a DFA equivalent to the NFA M = ({a,b,c,d}, {0,1}, δ, a, {b,d}) where δ is a defined as: (8) (Nov/Dec 2014)

δ	0	1
a	**{b,d}**	**{b}**
b	**c**	**{b,c}**
c	**d**	**a**
d	**--**	**a**

10. Prove the following by the principle of induction: (8)

$$\sum_{k=1}^{n} K^2 = \frac{n(n+1)(2n+1)}{6}$$

(May/June 2014)

11. Construct a DFA that accepts all strings on {0,1} except those containing the substring 101. (8) (May/June 2014)

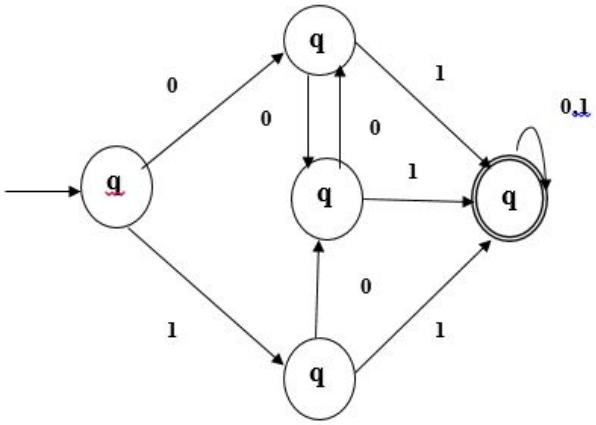
12. Construct a NFA accepting the set of strings over {a,b} ending in *aba*. Use it to construct a DFA accepting the same set of strings. (8) (May/June 2014)

13. Construct a DFA through NFA with ε moves which accepts a language consisting the strings of any number of a's followed by any number of b's followed by any number of c's. (8) (May/June 2014)

14. Design a finite automaton for the regular expression $(0+1)^*(00+11)(0+1^*)$ (8) (May/June 2014)

15. Find the regular expression of a language that consist of set of string starts with 11 as well as ends with 00 using Rij formula (May 2015)

16. Explain the DFA Minimization algorithm and Minimize the finite automaton show in figure below and show both the given and reduced one are equivalent. (May/June 2014)


q
q
q
q
q
0
0
0
1
1
0,1
0
1
1

UNIT II
SIMPLE 4 MARK QUESTION WITH ANSWER

1. **What do you mean by null production and unit production? Give an example. (May/June 2016)**

   A production which is of the form A→ ε is called null production

   A unit production is a production which is of the form A→B, where A and B are variable

2. **Construct a CFG for the set of strings that contain equal number of a's and b's over ∑={a,b}. (May/June 2016)**

   Consider Grammar G = {V,T,P,S}

   V = (S,a,b) T = (S) P = {S →aSb , S →bSa , S → ab/ba} S = {S}

3. **Define the term parse tree. (Nov/Dec 2015) (May/June 2012)**

   The parse tree for G = (V,T,P,S) are tree with following condition.

   1) Each interior node is labeled by a variable in V
   2) Each leaf is labeled by either a variable, a terminal or ∈
   3) If interior node is labeled A, and its children are labeled as X1,X2, .. Xk, where A → X1 X2.. Xk

   Example:

   E ⇒ E * (E)

4. **What is meant by ambiguity in grammars? (Nov/Dec 2015) (May/June 2013)**

   A grammar is said to be ambiguous if it has more than one derivation trees for a sentence or in other words if it has more than one leftmost derivation or more than one rightmost derivation.

5. **Define the term Chomsky Normal Form. (Nov/Dec 2015), (Nov/Dec 2012)**

   In formal language theory, a context-free grammar is in Chomsky Normal form (CNF) if the right-hand sides of all production rules start with a terminal symbol with following form:

   Eg:

   (1) S →BA (2) S →a

   Here A,B,C are variable and a is a terminal. Further G has no useless symbols.

6. **Construct the context free grammar representing the set of palindromes over (0+1)*. (Nov/Dec 2015) (May/June 2014)**

   Consider Grammar G = {V,T,P,S}

   V = (S,∈0,1) T = (S) P = {S → 0 | 1 | ∈, S → 0S0 , S → 1S1} S = {S}

7. **Let G be the grammar with**

   S →aB/bA

A→a/aS/bAA

B→b/bS/aBB

For the string aaabbabbba, find the leftmost derivation.(Nov/Dec 2015)

S→aB

S→aaBB

S→aaaBBB

S→aaabBB

S→aaabbB

S→aaabbaBB

S→aaabbabB

S→aaabbabbS

S→aaabbabbbA

S→aaabbabbba

8. **What is a CFG (May/June 2013)**

   A Context Free Grammar G = (V,T,P,S) consists of vocabulary V, a subset T of V consisting of terminal elements, a start symbol S from V and a set of production P. Every production in P must contain atleast one non-terminal (N = V-T) on its left side.

9. **What is meant by Greibach Normal Form (May/June 2013)**

   In formal language theory, a context-free grammar is in Greibach normal form (GNF) if the right-hand sides of all production rules start with a terminal symbol, optionally followed by some variables (Terminal or non-terminal).

   GNF Form:

   $A \rightarrow a\alpha$ where A is variable, a is terminal and α is string of variables

10. **Write the CFG for the language $L = \{a^n b^n \mid n \geq 1\}$ (Nov/Dec 2013)**

    Consider Grammar G = {V,T,P,S}

    V = (S,a,b) T = (S) P = {S →aSb , S → ab} S = {S}

11. **What are useless symbols in a grammar ? (Nov/Dec 2007)**

    A Symbol X is useful for a grammar G = (V,T,P,S) if there is some derivation of the form

    $S \overset{*}{\Rightarrow} \alpha X \beta \overset{*}{\Rightarrow} w$, where w is in Terminal. If X is not useful we say it useless.

    Ex:

    G = (V,T,P,S) where V = (S,T,X) T = (0,1)

    S → 0T | 1T | X | 0 | 1 (RULE 1)

    T → 00 (RULE 2)

Here X is an useless symbol has it does not have any useful production

12. List out the Types of Grammar?

According to Noam Chomsky the types of grammar are

Type 0 – Unrestricted grammar

Type 1 – Context Sensitive grammar

Type 2 – Context free grammar

Type 3 – Regular Grammar

13. What is Sentential Forms ?

Derivation from the start symbol produced string are called "Sentential form"
Eg:

If $S \overset{*}{\Rightarrow} \alpha$ is a sentential form

14. What is Recursive inference

It is an approach for inferring whether the given string belong to the given CFG. Here the rules are formed from the body to head. This procedure is called as recursive inferences.

15. What is leftmost and rightmost derivatives

In order to restrict the number of choices we have in deriving a string, it is often useful to require that at each step we replace the leftmost variable by one of its production bodies. Such a derivation is called a leftmost derivation, and we indicate that a derivation is leftmost by using the relation $\underset{lm}{\Rightarrow}$. Similarly it is possible to require that at each step the rightmost variable is replaced by one of its bodies is called as rightmost derivation and it is represented by the symbol $\underset{rm}{\Rightarrow}$.

16. What is recursively enumerated language

The Type 0 unrestricted language is called as recursively enumerated language. It include all formal grammars. They generate exactly all language that can be recognized by Turing Machine.

17. If S →aSb | aAb , A → bAa , A → ba .Find out the CFL

soln.
S→ aAb ⇒abab (sub A→ba)
S→aSb ⇒a aAb b ⇒a a ba b b (sub S→aAb & A→ba)
S→aSb ⇒ a aSb b ⇒a a aAb b b ⇒a a a ba b bb (S→aSb, S→aAb & A→ba)
Thus L={$a^n b^m a^m b^n$, where n,m≥1}

18. What are the applications of Context free languages?

Context free languages are used in :

- ➢ Defining programming languages.
- ➢ Formalizing the notion of parsing.

- ➢ Translation of programming languages.
- ➢ String processing applications.

PART B/C

Theory questions

1. **Discuss the following:**
   (i)CFG and Parse trees
   (ii) Ambiguity in Context Free Grammar with example (16)(Nov/Dec 2015)

2. **Let G =(V,T,P,S) be a context free grammar then prove that if the recursive inference procedure tells us that terminal string W is in the language of variable A, then there is a parse tree with root A and yield w. (10) (Nov/Dec 2015)**

3. **Explain the Type of Grammar in detail with proper examples (6) (Nov/Dec 2015)**
4. **What is an ambiguous grammar? Explain with an example. (6) (Nov/Dec 2015), (May/June 2016)**

Normal Form question

5. **Define the two normal forms that are to be converted from a context free grammar(CFG). Convert the following CFG to Chomsky normal form:**
   S→A/B/C
   A→aAa/B
   B→bB/bb
   C→baD/abD/aa
   D→ aCaa/D (10)(May/June 2016)

6. **Construct the following grammar in CNF:**
   S→ ABC/BaB
   A →aA/BaC/aaa
   B →bBb/a/D
   C →CA/AC
   D→ Є (8)(Nov/Dec 2015)

7. **Construct a equivalent grammar G in CNF for the grammar G1 where**
   G1=({S,A,B},{a,b},{S →ASB/Є , A→ aAS/a,B→ SbS/A/bb},S) (10) (Nov/Dec 2015)

8. **Convert the following CFG G to Greibach normal form generating the same language**
   S→ABA
   A→aA/λ
   B→bB/λ (6) (May/June 2016)

9. **What is the purpose of normalization? Construct the CNF and GNF for the following grammar and explain the steps.**
   S→aAa/bBb/Є
   A→C/a
   B→C/b
   C→CDE/Є
   D→A/B/ab (10)(May/June 2016)

10. **Construct a reduced grammar equivalent to the grammar G=(N,T,S,P) where**
    N={S,A,C,D,E}
    T={a,b}
    P={ S→aAa,A→Sb/bCC/DaA,C→abb/DD,D→aDA,E→aC } (6) (May/June 2016)

11. **Convert the following grammar into CNF**
    S → cBA
    S → A
    A → cB | AbbS
    B → aaa (8)
12. **Convert the following grammar into GNF**
    S → XY1 | 0
    X → 00X | Y
    Y → 1X1 (8)
13. **Construct a equivalent grammar G in CNF for the grammar G1 where**
    G1 =({S,A,B},{a,b},{S->bA/aB,A->bAA/aS/a, B->aBB/bS/b},S) (8)

<u>Ambiguity and Parse tree</u>

14. **Given the grammar G=(V,∑,R,E), where V={E,D,1,2,3,4,5,6,7,8,9,0,+,-,*,/,(,)}, ∑={1,2,3,4,5,6,7,8,9,0,+,-,*,/,(,)}, and R contains the following rules:**
    E D/(E)/E+E/E-E/E*E/E/E
    D 0/1/2/3/4/5/6/7/8/9
    Find a parse tree for the string 1+2*3 (6) (Nov/Dec 2015)

15. **Consider the following grammar:**
    E→ E+T/T
    T →T*F/F
    F →(E)/ id
    (i) Give a rightmost derivation and leftmost derivation for the sentence w=id*(id+id)*id
    (ii) Is the above grammar ambiguous? Justify.
    (iii) Construct the parse tree for the sentence w=id*(id+id)*id (16)(Nov/Dec 2015)

16. **Show the derivation steps and construct derivation tree for the string ababbb by using leftmost derivation with the grammar**
    S→AB/Є
    A→aB
    B→Sb (5) (May/June 2016)

17. **Construct a CFG for the regular expression (011+1)(01) (6) (May/June 2016)**

18. **Consider the following grammar for list structures:**
    S → a | ^ | (T)
    T → T,S | S
    Find the left most derivation, right most derivation and parse tree for (((a,a),^(a)),a) (8)

19. **Explain about Parse trees. For the following grammar**
    S → aB| bA
    A → a|aS|bAA
    B → b|bS|aBB

For the string aaabbabbba. Find left most derivation, right most derivation and parse tree (8)

20. If G is the grammar S → SbS | a show that G is ambiguous (5)

UNIT III
SIMPLE 4 MARK QUESTION WITH ANSWER

1. What are the different ways of language acceptances by a PDA and define them ? (Dec 15)

PDA accepts its input either by "acceptance by final state" or "Acceptance by Empty stack"

PDA acceptance by empty stack method	**PDA acceptance by empty final method**
PDA accepts when set of strings that cause the PDA to empty its stack	PDA accepts its input by consuming it and entering an accepting state
For each PDA $P = (Q,\Sigma,\Gamma,\delta,q_0,Z_0,F)$ then $L(P) = \{w \mid (q_0,w,z_0) \quad (q,\in,\alpha)\}$	For each PDA $P = (Q,\Sigma,\Gamma,\delta,q_0,Z_0,F)$ then $N(P) = \{w \mid (q_0,w,z_0) \quad (q,\in,\in)\}$

2. Does a PDA has a memory? Justify (May 16)

The Push down automaton is in essence a nondeterministic fine automaton with ε-transitions permitted and one additional capability: a stack on which it can store a string of "stack symbols". Thus the automaton can remember an infinite amount of information.

3. Define pumping lemma for CFL (May 15,16)

Let L be a CFL+ then there exists a constant n such that if z is any string in L such that |z| is at least n, then we can write z = uvwxy subject to the following consitions:

- $|vwx| \leq n$. That is, the middle portion is not too long.
- $vx \neq \in$. Since v and x are the pieces to be pumped, these strings should not be empty
- For all $i \geq 0$, uv^iwx^iy is in L. This two strings are pumped any number of times

4. Differentiate PDA acceptance by empty stack method with acceptance by final method (May 15)

PDA acceptance by empty stack method	**PDA acceptance by empty final method**
PDA accepts when set of strings that cause the PDA to empty its stack	PDA accepts its input by consuming it and entering an accepting state

For each PDA $P = (Q,\Sigma,\Gamma,\delta,q_0,Z_0,F)$ then L(P)	For each PDA $P = (Q,\Sigma,\Gamma,\delta,q_0,Z_0,F)$ then N(P)
$= \{w \mid (q_0,w,z_0) \quad (q,\in,\alpha)\}$	$= \{w \mid (q_0,w,z_0) \quad (q,\in,\in)\}$

5. **Define Push Down Automata (May 2016, Dec 15)**

   The formal notation for PDA involves seven components the specification are :

   $P = (Q,\Sigma,\Gamma,\delta,q_0,Z_0,F)$ where,

   Q = Finite set of states, ,Σ = Finite set of input symbol,Γ = A finite stack alphabets,δ = transition function,q_0 = Start State,Z_0 = Start symbol, F = Accepting state

6. **Compare NFA and PDA**

NFA	PDA
The language accepted by NFA is the regular language	The language accepted by PDA is Context free language.
NFA has no memory.	PDA is essentially an NFA with a stack (memory).
It can store only limited amount of information.	It stores unbounded limit of information.
A language/string is accepted only by reaching the final state.	It accepts a language either by empty Stack or by reaching a final state.

7. **Define Deterministic PDA**

   A PDA $M = (Q,\Sigma,\Gamma,\delta,q_0,Z_0,F)$ is deterministic if and only if the following condition are meet.

   - $\delta(q,a,X)$ has at most one member for any q in Q, a in Σ or a =$\in$ and X in Γ
   - If $\delta(q,a,X)$ is nonempty, for some a in Σ, then $\delta(q,\in,X)$ must be empty

   Eg: $L = \{0^n 1^n \mid n \geq 1\}$

8. **What is the significance of PDA?**

   Finite Automata is used to model regular expression and cannot be used to represent non regular languages. Thus to model a context free language, a Pushdown Automata is used.

9. **Define Instantaneous description(ID) in PDA.**

PDA adopt "turnstyle" notation "$\vdash$" for ID . Let P = $(Q,\Sigma,\Gamma,\delta,q_0,Z_0,F)$ then $\vdash_P$ or $\vdash$ define ID of P. Suppose $\delta(q,a,X)$ contains (p,α) Then for all strings w in Σ^* and β in Γ^*:

$$(q,aw,X\beta) \vdash (p,w,\alpha\beta)$$

10. Give an example of a language accepted by a PDA but not by DPDA.

The language L = $\{ww^R, \mid w \text{ is in } (0+1)^*\}$ is a example of PDA which cannot be deterministic

11. construct PDA that accepts the language generated by the grammar S->aSbb|aab

The PDA P = {{q}, {a,b}, {S,a,b} , δ , q, S} Where δ defined by:

i) $\delta(q,\in,S) = \{(q,aSbb), (q,aab)\}$
ii) $\delta(q,a,a) = \{(q, \in)\}$
iii) $\delta(q,b,b) = \{(q, \in)\}$

12. construct PDA that accepts the language generated by the grammar S->0BB, B->0S|1S|0

The PDA P = {{q}, {0,1}, {S,B,0,1} , δ , q, S} Where δ defined by:

i) $\delta(q,\in,S) = \{(q,0BB)\}$
ii) $\delta(q,\in,B) = \{(q,0S), (q,1S), (q,0)\}$
iii) $\delta(q,0,0) = \{(q, \in)\}$
iv) $\delta(q,1,1) = \{(q, \in)\}$

13. What is the specification of PDA P_F which accept the language by final state which is also accepted by Empty stack PDA P_N

The specification of P_F is as follow:
$P_F = \{Q \cup \{p_0,p_f\}, \Sigma, \Gamma \cup \{X_0\}, \delta_F, p_0, X_0, \{p_f\}\}$ where
p_0 is the new state which push Z_0 then enter the start state of P_N q_0
p_f is the new accepting state of P_F

14. What is the specification of PDA P_N which accept the language by empty stack which is also accepted by final state PDA P_F

The specification of P_N is as follow:
$P_N = \{Q \cup \{p_0,p_f\}, \Sigma, \Gamma \cup \{X_0\}, \delta_N, p_0, X_0, \{p\}\}$ where
p_0 is the new state which push Z_0 then enter the start state of P_F q_0
For each accepting state of P_F add a transition on $\in$ to p, which pop it stack contents

15. What is the advantage of Pumping lemma over CFL

Pumping lemma over context free language used to check whether the given language are regular or CFL are not.

16. Construct a PDA for the given grammar S→ aSa| bSb |c (May '15)

17. Convert the following CFG to PDA S→ aAA , A→ aS|bS|a (Dec 15)

18. **Draw the pictorial notation of PDA which recognizes the language $\{0^n1^n \mid n>=0\}$**

PART B/C

Problematic question

1. **Construct a PDA to accept the following language L on $\sum$={a,b} by empty stack . L ={ww^R | W Є $\sum^+$} (Dec '15 May '16 10 mark)**
2. **Construct a PDA to accept the language L = $\{a^n b^n c^n \mid n \geq 1\}$ by empty stack and by final state (June '14 10 marks)**
3. **Give formal PDA that accepts {wcw^R | w in (0+1)*} by empty stack (Dec '13 8 marks)**
4. **Design a PDA to accept $\{0^n1^n \mid n > 1\}$ Draw the transition diagram for the PDA. Show by instantaneous description that the PDA accepts the string '0011' (Dec '15 10 mark)**
5. **Construct PDA to recognize the Grammar G with following production and trace for a String of acceptance and rejection (May '16 10 mark)**

   S → aSA / ε

   A → bB /cc

   B →bd / ε
6. **Convert PDA to CFG. PDA is given by (Dec '15 10 mark)**

   P = ({p,q}, {0,1}, {X,Z}, δ , q, Z) where δ is given by

   δ(p,1,Z) = {(p,XZ)}

   δ(p,ε,Z) = {(p,ε)}

   δ(p,1,X) = {(p,XX)}

   δ(q,1,X) = {(q, ε)}

   δ(p,0,X) = {(q,X)}

   δ(q,0,Z) = {(p,Z)}

7. **Let M = ($\{q_0,q_1\}$, {0,1}, $\{x,z_0\}$, δ , q_0, z_0,φ) where δ is given by**

   $\delta(q_0,0,z_0)$ = $\{(q_0,xz_0)\}$

   $\delta(q_0,0,x)$ = $\{(q_0,xx)\}$

   $\delta(q_0,\in,x)$ = $\{(q_1, \in)\}$

   $\delta(q_0,1,x)$ = $\{(q_1, \in)\}$

   $\delta(q_1,\in, z_0)$ = $\{(q_1, \in)\}$

   Construct a CFG for the PDAM (June '14 10 marks)

Theorem /Pumping Lemma and theoretical question

8. **What is an instantaneous description of a PDA ? How will you represent it ? Also give the three important principle of ID and their transaction (May '16 6 mark)**
9. **Explain acceptance by final state and acceptance by empty stack of a pushdown automata (May '16 8 mark)**
10. **State pumping Lemma for CFL. Use pumping lemma to show that the language L =$\{a^i b^j c^k$ | i < j< k} is not a CFL (May '16 8 mark)**
11. **Discuss on Deterministic PDA.Differentiate between deterministic pushdown automata and non deterministic pushdown automata (May '16 , Dec '15 6 mark)**
12. **State the pumping lemma for CFLs. What is its main application ? Give two example (Dec '11 8 marks)**
13. **If L is context free language prove that there exists a PDA M, such that L = N(M) (Dec '14 8 marks)**
14. **Prove that if L is $N(M_1)$ (the language accepted by empty stack) for some PDA M_1, then L is $N(M_2)$ (the language accepted by final state) for some PDA M_2 (Dec '14 8 marks)**
15. **State pumping Lemma for CFL. Use pumping lemma to show that the language L =$\{a^n b^n c^n$ | n ≥1} is not a CFL (Dec '15 6 mark)**

UNIT IV
PART A

1. **List the component of Turning machine (DEC 15)**
   A Turing machine has two main components:
   - A finite state machine that acts as a controller
   - A two way infinite tape divided into cells, from which the machine can read input symbols, moves left or right, and onto which the machine can write output symbols
2. **State the advantage of Turning machine over automata (June 16)**
   - The simplest automata used for computation is a finite automaton. It can compute only very primitive functions; therefore, it is not an adequate computation model.
   - The Turing machine can be thought of as a finite automaton or control unit equipped with an infinite storage (memory).
3. **Differentiate FA and TM (June 16)**

Finite Automata	Turning Machine
FA is denoted by 5 tuple $(Q, \Sigma, \delta, q_0, F)$	A Turing machine is denoted by 7 tuple $M=(Q, \Sigma, \Gamma, \delta, q0, B, F)$
It has no memory unit	A TM has infinite storage capacity

It can recognize context free language	It can compute integral functions

4. Define Turing machine. (DEC 15, June 16)

A Turing machine is denoted as M=(Q, Σ, Γ ,δ ,q0, B,F)

Q is a finite set of states.

Σ is set of i/p symbols ,not including B.

Γ is the finite set of tape symbols.

q0 in Q is called start state.

B in Γ is blank symbol.

F is the set of final states.

δ is a mapping from Q X Γ to Q X Γ X {L,R}.

5. Define Instantaneous description of TM.

The ID of a TM M is denoted as α1 q α2 . Here q is the current state of M is in Q; α1 α2 is the string in Γ * that is the contents of the tape up to the rightmost nonblank symbol or the symbol to the left of the head, whichever is the rightmost.

6. What are the applications of TM?

TM can be used as:

- Recognizers of languages.
- Computers of functions on non negative integers.
- Generating devices.

7. What is the basic difference between 2-way FA and TM?

Turing machine can change symbols on its tape , whereas the FA cannot change symbols on tape. Also TM has a tape head that moves both left and right side ,whereas the FA doesn't have such a tape head.

8. Define a move in TM.

Let $X_1 X_2 \ldots X_{i-1} q X_i \ldots X_n$ be an ID.

The left move is: if $\delta (q, X_i) = (p, Y, L)$,if i>1 then

$X_1 X_2 \ldots X_{i-1} q X_i \ldots X_n \vdash X_1 X_2 \ldots X_{i-2} p X_{i-1} Y X_{i+1} \ldots Xn.$

The right move is if $\delta (q, Xi) = (p, Y, R)$,if i>1 then

$X_1 X_2 \ldots X_{i-1} q X_i \ldots X_n \vdash X_1 X_2 \ldots X_{i-1} Y p X_{i+1} \ldots X_n.$

9. What is the language accepted by TM?

The language accepted by M is L(M) , is the set of words in Σ * that cause M to enter a final state when placed ,justified at the left on the tape of M, with M at q_o and the tape head of M at the leftmost cell. The language accepted by M is:

{ w | w in Σ * and q_0 w |-- α1 p α2 for some p in F and α1 ,α2 in Γ * }.

10. What are the various representation of TM?

We can describe TM using:

- Instantaneous description.
- Transition table.
- Transition diagram.

11. What are the possibilities of a TM when processing an input string?

- TM can accept the string by entering accepting state.
- It can reject the string by entering non-accepting state.
- It can enter an infinite loop so that it never halts.

12. What are the techniques for Turing machine construction?

• Storage in finite control.

• Multiple tracks.

• Checking off symbols.

• Shifting over

• Subroutines.

13. What is the storage in FC?

The finite control(FC) stores a limited amount of information. The state of the Finite control represents the state and the second element represent a symbol scanned.

14. When is checking off symbols used in TM?

Checking off symbols is useful method when a TM recognizes a language with repeated strings and also to compare the length of substrings.

(eg) : { ww | w Є Σ * } or {aibi | i>=1}.

This is implemented by using an extra track on the tape with symbols Blank or √.

15. What is a multihead TM?

A k-head TM has some k heads. The heads are numbered 1 through k, and move of the TM depends on the state and on the symbol scanned by each head. In one move, the heads may each move independently left or right or remain stationary.

16. What is a 2-way infinite tape TM?

In 2-way infinite tape TM, the tape is infinite in both directions. The leftmost square is not distinguished. Any computation that can be done by 2-way infinite tape can also be done by standard TM.

17. Differentiate PDA and TM.

PDA	TM
1. PDA uses a stack for storage.	1. TM uses a tape that is infinite .
2.The language accepted by PDA is CFL.	2. Tm recognizes recursively enumerable languages.

18. What is a multi-tape Turing machine? (DEC 15)

A multi-tape Turing machine consists of a finite control with k-tape heads and k-tapes ; each tape is infinite in both directions. On a single move depending on the state of finite control and symbol scanned by each of tape heads ,the machine can change state print a new symbol on each cells scanned by tape head, move each of its tape head independently one cell to the left or right or remain stationary.

19. What is a multidimensional TM?

The device has a finite control , but the tape consists of a k-dimensional array of cells infinite in all 2k directions, for some fixed k. Depending on the state and symbol scanned , the device changes state , prints a new symbol and moves its tape-head in one of the 2k directions, either positively or negatively ,along one of the k-axes.

20. State the halting problem of TMs.

The halting problem for TMs is: Given any TM M and an input string w, does M halt on w .This problem is undecidable as there is no algorithm to solve this problem.

PART B

Problematic Question

1. Construct a TM to accept palindrome in an alphabet set $\sum = \{a,b\}$. Trace the string "baab" and "abab" (8 mark May 16)
2. Design a Turing Machine to reorganization the language L= $\{a^n c b^n | n \geq 0\}$ (12 mark May 16)
3. Explain the programming technique for TM (10 mark Dec 15)
4. Design a Turing Machine to reorganization the language L= $\{0^n 1^n | n \geq 1\}$ also specify the ID to trace the string 0011(10 mark Dec 15,June 14)
5. Describe how TM is useful for computing arithmetic functions. How can we perform proper subtraction? (8)
6. Design a Turing Machine M to implement the function "multiplication" using the subroutine 'copy'. (12 mark Dec 14)

Theory Question

7. Explain the variations of TM (16 mark May 16, Dec13)

(or)

Describe the following Turing machine and their working. Are they more powerful than the Basic Turing Machine?

- Multi-tape (Multiple Track) Turing Machine
- Multi-Dimensional Turing Machine
- Two-Way infinite tape TM

8. Explain halting problem. Is it solvable or unsolvable problem discuss. (8 mark May 16)

(or)

State and describe halting problem. For TM (6 mark Dec 15)

9. Describe Chomsky hierarchy of language with example. What are the devices that accept these language. (8 mark May 16, Dec 15)

What is the role of checking off symbols in a Turing Machine? (6 Mark June 14)

UNIT V
PART A

1. **Define Diagonlization (L_d) Language (Dec '14)**
   Ans:
   The diagonlization language is the set of string w_i, such that w_i is not in $L(M_i)$. L_d consists of all string w such that the TM M whose code is w does not accept when given w as input.
   $$L_d = \{w_i \mid w_i \text{ is not accept by } M_i\}$$
2. **Give examples for NP complete problems (Dec '14)**
   Ans:
   Traveling Salesman problem, 0/1 knapsack problem, Hamiltonian circuit problem and graph coloring problem.
3. **Define - RE (Recursive Enumerable language) (May 15)**
   Ans:
   A recursively enumerable language is a formal language for which there exists a Turing machine (or other computable function) which will enumerate all valid strings of the language. Recursively enumerable languages are known as type-0 languages in the Chomsky hierarchy of formal languages. All regular, context-free, context-sensitive and recursive languages are recursively enumerable
4. **Define recursive and non-recursive language (May '15)**
   Ans:
   A formal language is recursive if there exists a total Turing machine that, when given a finite sequence of symbols as input, accepts it if belongs to the language

and rejects it otherwise. Recursive languages are also called decidable. The Undecidable language is called as non-recursive language.

5. **State Rice theorem**

   Ans:

   Rice's theorem states that, for any non-trivial property of the recursive enumerable languages is undecidable.

6. **Distinguish between time and space complexities**

   Ans:

Time Complexities	Space Complexities
Time complexity deals with finding out how the computational time of an algorithm changes with the change in size of the input.	Space complexity deals with finding out how much (extra)space would be required by the algorithm with change in the input size.
To calculate time complexity of the algorithm the best way is to check if we increase in the size of the input, will the number of comparison(or computational steps) also increase	To calculate space complexity the best bet is to see additional memory requirement of the algorithm also changes with the change in the size of the input.

7. **What are tractable problem**

   Ans:

   Tractable problems can be solved by computer algorithms that run in polynomial time; i.e., for a problem of size n, the time or number of steps needed to find the solution is a polynomial function of n

   Ex: Searching in ordered or unordered list, sorting ect.,

8. **What is an intractable problem ? Give examples**

   Ans:

   Algorithms for solving hard, or intractable, problems, requires time that are exponential functions of the problem size.

   Ex: Tower of Hanoi, List all permutation of n number, ect.

9. **Define NP complete problem**

   Ans:

   In computational complexity theory, a decision problem is NP-complete when it is both in NP and NP-hard. NP may be equivalently defined as the set of decision problems that can be solved in polynomial time on a non-deterministic Turing machine.

10. **What are recursive sets (DEC '13)**

    Ans:

In computability theory, a set of natural numbers is called recursive, computable or decidable if there is an algorithm which terminates after a finite amount of time and correctly decides whether or not a given number belongs to the set.

11. What is universal turning machine (Dec '13)

A universal Turing machine ,is a Turing machine that can simulate an arbitrary Turing machine on arbitrary input. The universal machine essentially achieves this by reading both the description of the machine to be simulated as well as the input thereof from its own tape.

12. List the Basic Primitive function

Ans:

Zero function : $C^n : N^n \rightarrow N$, $C^n(x1,----x2) = 0$

Projection function: $P^n_i = N^n \rightarrow N$, $P^n_i(xi,----xn) = xi$

Successor function: $S: N \rightarrow N$, $S(x) = x+1$

13. Define Universal Language

Ans:

The Universal language L_u consists of strings that are interpreted as a TM followed by an input for that TM. The string is in L_u if the TM accepts that input.

14. Define Gödel numbering

Ans:

A Gödel numbering is a function that assigns to each symbol of some formal language a unique natural number, called its Gödel number. A Gödel numbering can be interpreted as an encoding in which a number is assigned to each symbol of a mathematical notation, after which a sequence of natural numbers can then represent a sequence of symbols. Gödel used a system based on prime factorization. Given sequence (x1,x2,x3,----xn)

$$\mathbf{enc(x1,x2,x3,....xn) = 2^{x1}.3^{x2}.5^{x3}.------P^{xn}}$$

15. Give two property of RE sets which are undecidable

Ans:

The language accepted by a TM is called Recusively enumerable (RE) language. Every non-Trivial property of RE is undecidable. Here are two such properties

(a) Whether the language accepted by a TM is regular language

(b) Whether the language accepted by a TM is context free language

These are the two properties that are undecidable

16. Is travelling salesman problem a NP or P Problem ? Justify

Ans:

Travelling salesman problem (TSP) is NP complete. It can be proved by reducing travelling salesman problem to Hamiltonian circuit. As Hamiltonian circuit is NP complete we can easily prove that TSP is not NP Complete.

PART B

1. (a) Show that the union of two recursive language is recursive and union of two recursively enumerable language is recursive 12 (DEC 14)
   (b) Define the language L_u and show that L_u is RE language (4 mark Dec'14)
2. State and prove post correspondence problem and Give example (16 Mk. Dec'14)

   (or)

   Explain PCP and deicdable and Undecidable problem with example
3. (a) Explain the class P and NP problems (May '15)
   (b) Write short notes on Universal Turing machine (8 Mark)
4. Prove that a problem P2 cannot be solved in polynomial time can be proved by the reduction of a problem P1 which is under class P1 to P2
5. (a) Prove that if a language is recursive if and only if it and its complement are both recursively enumerable (8 Mark Dec '13)
   (b) Prove L_{ne} is recursively enumerable (8 Mark June '13)
6. Define diagonalization language. Show that the language Ld is not a recursively enumerable language (June '14)
7. Define PCP. Let sigma = {0,1}. Let A and B be the lists of three strings each defined as :

	List A	List B
i	w_i	x_i
1	1	111
2	10111	10
3	10	0

   Dose this PCP have a solution