CS 3452 - THEORY OF COMPUTATION

UNIT I AUTOMATA AND REGULAR EXPRESSIONS

Need for automata theory - Introduction to formal proof – Finite Automata (FA) – Deterministic Finite Automata (DFA) – Non-deterministic Finite Automata (NFA) – Equivalence between NFA and DFA – Finite Automata with Epsilon transitions – Equivalence of NFA and DFA- Equivalence of NFAs with and without ε-moves- Conversion of NFA into DFA – Minimization of DFAs.

PART-A

1. Define hypothesis. R

The formal proof can be using deductive proof and inductive proof. The deductive proof consists of sequence of statements given with logical reasoning in order to prove the first or initial statement. The initial statement is called hypothesis.

2. Define inductive proof. R **Nov/Dec 2010**
(Or) State the principle of induction **Nov/Dec 2012**

This is a very powerful and important technique for proving theorems.

For each positive integer n, let P(n) be a mathematical statement that depends on n. Assume we wish to prove that P(n) is true for all positive integers n.

A proof by induction of such a statement is carried out as follows:

Basis: Prove that P(1) is true.

Induction step: Prove that for all n≥1, the following holds: If P(n) is true, then P(n + 1) is also true.

In the induction step, we choose an arbitrary integer n≥1 and assume that P(n) is true; this is called the induction hypothesis. Then we prove that P(n + 1) is also true.

3. What is structural induction? R **May/June 2011**

To prove a property of the elements of a recursively defined set, we use structural induction.

Basis Step: Show that the result holds for all elements specified in the basis step of the recursive definition.

Inductive Step: Assume that the property holds for the elements currently in the recursively defined set.
Show that it is true for each of the rules used to construct new elements in the recursive step of the definition.

4. What is proof by contradiction? R **May/June 2012**

The method of proof by contradiction is to assume that a statement is not true and then to show that that assumption leads to a contradiction. A good example of this is by proving that $\sqrt{2}$ is irrational.

5. Define deductive proof. **R** **Nov/Dec 2014**

Deductive proof consists of a sequence of statements whose truth leads from some initial statement, called the hypothesis to a conclusion statement. Each step in the proof must follow some accepted logical principle, from either the given facts or some previous in the deductive proof or a combinations of these.

6. Define Set, Infinite and Finite Set. **R**

Set is Collection of various objects. These objects are called the elements of the set.

Eg : A = { a, e, i, o, u }

Infinite Set is a collection of all elements which are infinite in number.

Eg: A = {a | a is always even number}

Finite Set is a collection of finite number of elements. **Eg : A = { a, e, i, o, u }**

7. Give some examples for additional forms of proof. **U**

1. Proofs about sets
2. Proofs by contradiction
3. Proofs by counter examples.

8. Prove 1+2+3+………………+n= n(n+1)/2 using induction method. A

Consider the two step approach for a proof by method of induction

1. Basis of induction:

Let n = 1 then LHS = 1 and RHS = 1 + 1 / 2 = 1 Hence LHS = RHS.

2. Induction hypothesis:

To prove 1 + 2 + 3 …… + n = n (n + 1) / 2 + (n + 1)

Consider n = n + 1

then 1 + 2 + 3 ……+ n + (n + 1) = n (n + 1) / 2 + (n + 1)

= n2 + 3n + 2 / 2

= (n + 1) (n + 2) / 2

Thus it is proved that 1 + 2 + 3 …… + n = n (n + 1) / 2

9. Write down the operations on set. U

i) A U B is Union Operation

If A = { 1, 2, 3 } B = { 1, 2, 4 } then A U B = { 1, 2, 3, 4 }

i.e. combination of both the sets.

ii) A ∩ B is Intersection operation

If A = { 1, 2, 3 } B = { 1, 2, 4 } then A U B = { 2, 3 }

i.e. Collection of common elements from both the sets.

iii) A – B is the difference operation

If A = { 1, 2, 3 } B = { 1, 2, 4 } then A - B = { 3 }

i.e. elements which are there in set A but not in set B.

10. Write any three applications of Automata Theory. U

1. It is base for the formal languages and these formal languages are useful of the programming languages.
2. It plays an important role in complier design.
3. To prove the correctness of the program automata theory is used.
4. In switching theory and design and analysis of digital circuits automata theory is applied.
5. It deals with the design finite state machines.

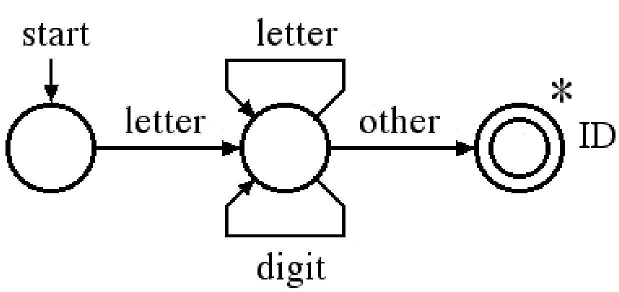
11. Define i. Finite automaton (or) What is a finite automaton? R
ii. Transition diagram May/June 2013, Nov/Dec 2012,2015 & 2017

FA consists of a finite set of states and a set of transitions from state to state that occur on input symbols chosen from an alphabet ∑. Finite Automaton is denoted by a 5- tuple (Q,∑,δ,q0,F), where Q is the finite set of states , ∑ is a finite input alphabet, q0 in Q is the initial state, F is the set of final states and δ is the transition mapping function Q * ∑ to Q.

Two types: **Deterministic Finite Automata (DFA)**
Non-Deterministic Finite Automata (NFA)

Transition diagram is a directed graph in which the vertices of the graph correspond to the states of FA. If there is a transition from state q to state p on input a, then there is an arc labeled ‘a ‘from q to p in the transition diagram.

12. Draw transition diagram for an identifier. A Nov/Dec 2013

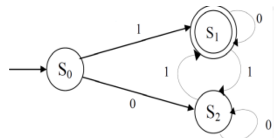


13. Define Deterministic Finite Automata. R May/June 2013, Nov-Dec 2016,2019

The finite automata are called DFA if there is **only one path for a specific input from current state to next state.**

A finite automata is a collection of 5 tuples (Q, Σ. δ, q0, F)
where Q is a finite set of states, which is non-empty.
Σ is a input alphabet, indicates input set.
δ is a transition function or a function defined for going to next state.

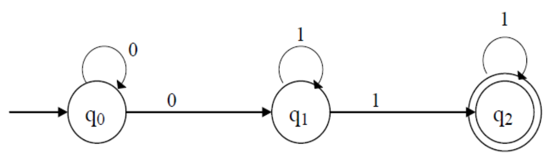
q0 is an initial state (q0 in Q)
F is a set of final states.



14. Define Non-Deterministic Finite Automata. R Nov/Dec 2013

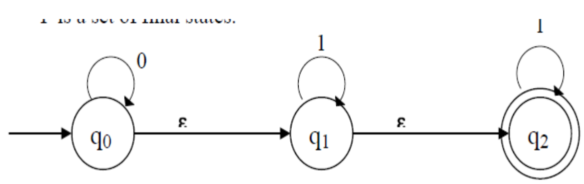
The finite automata are called NFA when there exists **many paths for a specific input from current state to next state.**

A finite automata is a collection of 5 tuples (Q, Σ. δ, q0, F)
where Q is a finite set of states, which is non-empty.
Σ is a input alphabet, indicates input set.
δ is a transition function or a function defined for going to next state.
q0 is an initial state (q0 in Q)
F is a set of final states.



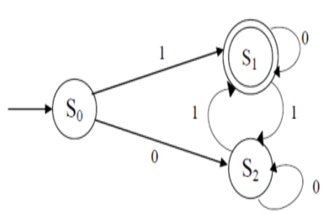
15. Define NFA with € transition. R

The € **is a character** used to indicate null string. i.e the string which is used simply for transition from **one state to other state without any input.**

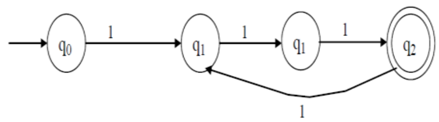


A Non Deterministic finite automata is a collection of 5 tuples (Q, Σ. δ, q0, F)
where Q is a finite set of states, which is non-empty.
Σ is a input alphabet, indicates input set.
δ is a transition function or a function defined for going to next state.
q0 is an initial state (q0 in Q)
F is a set of final states.

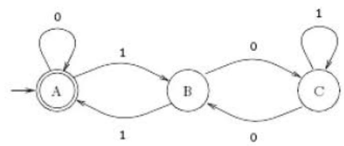
16. Design FA which accepts odd number of 1's and any number of 0's. C



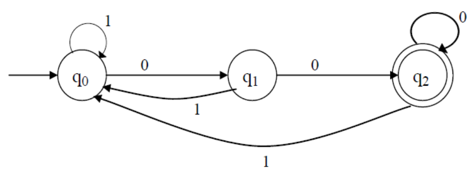
17. Design FA to check whether given unary number is divisible by three. C



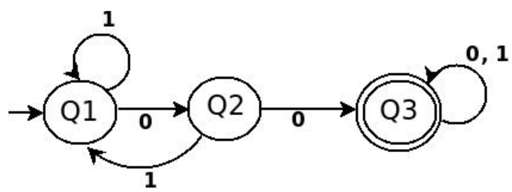
18. Design FA to check whether given binary number is divisible by three. C



19. Design FA to accept the string that always ends with 00. C

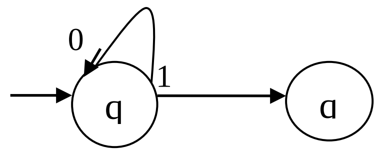
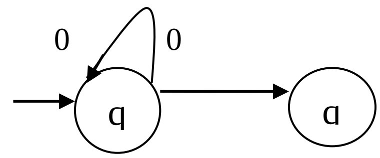


20. Design DFA to accept the strings over {0,1} with two consecutive 0's. C Nov/Dec 2014



21. State the difference between NFA & DFA. AN May/June 2011& May/June 2014 Nov/Dec 2018

S.NO	DFA	NFA
1	For each input symbol there is exactly one transition out of each state.	For each input symbol there is one or more transition from a state on the same input symbol.
2	It doesn't allow ξ moves	It allows ξ moves
3	1. $\delta(q,\xi) = q$ 2. $\delta(q,wa) = \delta(\delta`(q,w),a)$	1. $\delta(q,\xi) = q$ 2. $\delta(q,wa) = \delta(\delta`(q,w),a)$ 3. $\delta(p,x) = \bigcup_{q \text{ in } p} \delta(q,w)$
4	Every DFA can simulate as NFA	NFA can't simulate as DFA
5	Transition function mapping from $Q \times \Sigma$ to Q.	Transition function mapping from $Q \times \Sigma$ to 2^Q.

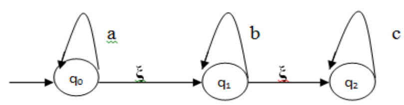
6	DFA stands for Deterministic finite automata.	NFA stands for Non deterministic finite automata.
7		

22. Define the term Epsilon(€) transition. **R** **May/June 2013**

The € **is a character** used to indicate null string. i.e the string which is used simply for transition from **one state to other state without any input.**

23. Define ξ –Closure (q) **with an example.** **R May/June 2012, Nov/Dec 2022**

ξ –Closure (q) defines the set of all vertices p such that there is path from q to p labeled ξ.

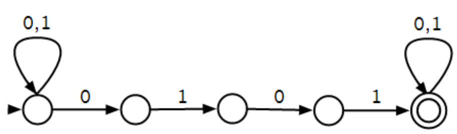


Solution:

ξ -**closure**(q_0) = {q_0,q_1,q_2}

ξ -**closure**(q_1) = {q_1,q_2}

24. Draw a NFA to accept strings containing the substring 0101. C **May-June 2016**



PART B

1. Prove the following by induction for all n≥0 A May/June 2014
   i. $1^2+2^2+3^2+4^2+\ldots\ldots.+n^2=(n(n+1)(2n+1))/6$ May/June 2016
   ii. $1^3+2^3+\ldots.+n^3=(n^2(n+1)^2)/4$
2. Prove the following by mathematical induction method: A
   i. $2^n>n$ for all $n\geq0$
   ii. $X\geq4, 2^x\geq x^2$
3. Prove by Induction that A
   i. $n^3+(n+1)^3+(n+2)^3$ is divisible by 3 and n>0
   ii. $S(n)=a^n-b^n$ is divisible by a-b for all n>0
4. Prove A
   i. $S(n)=5^{2n}-1$ is divisible by 24 for n>0
   ii. $1+2+\ldots\ldots+n=(n(n+1))/2$ **Nov/Dec 2022**

5. i. Design DFA to accept the Language C
L={w/w has both even number of 0's and even number of 1's}
ii. Construct DFA that accepts input string of 0's and 1's that end with 11.
Construct DFA for the set of all strings {0,1} with strings ending with 01.
Construct DFA for the Language $L=\{0^n/n \bmod 3=2, n\geq 0\}$
Construct DFA for all the set of strings with {0,1} that has three consecutive 1's.

6. i. Construct an NFA for the set of strings with {0,1} ending with 01 draw the transition table for the same and check whether the input string 00101 is accepted by above NFA. C
ii. Construct NFA for set of all strings {0,1} that ends with three consecutive 1's at its end. C
iii. Construct NFA for set of all strings {a,b} with abb as substring. C

7. If a Regular language 'L' is accepted by a Non – deterministic Finite automata then there exist a Deterministic Finite Automata that accepts 'L' A
Nov/Dec 2013&Nov/Dec 2014

8. A Language 'L' is accepted by some ε – NFA if and only if L is accepted by NFA without ε transition A May/June 2012 & Nov/Dec 2013

9. Convert to a DFA, the following NFA A May/June 2013

	a	b
p(start)	{p}	{p,q}
q	{r}	{r}
r	{ Φ }	{ Φ }

10. Convert the following NFA-with ε, to a NFA- without ε A

	0	1	2	ε
q_0 (start)	$\{q_0\}$	{φ}	{φ}	$\{q_1\}$
q_1	{φ}	$\{q_1\}$	{φ}	$\{q_2\}$
* q_2	{φ}	{φ}	{q2}	{φ}

11. Convert the following NFA-with ε, to a NFA- without ε A

	a	b	ε
q_0 (start)	$\{q_0\}$	{φ}	$\{q_1\}$
* q_1	{φ}	$\{q_1\}$	{φ}

12. Convert the following NFA-with ε, to a NFA- without ε A

	a	b	c	ε
p(start)	{q}	{p}	φ	φ
q	{r}	Φ	{q}	φ
*r	Φ	Φ	φ	{r}

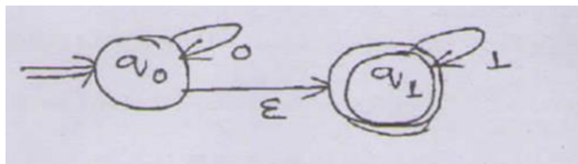
13. i. Prove that $\sqrt{2}$ is not rational. A

ii. Construct a DFA accepting all strings w over {0,1} such that the number of 1's in w is 3 mod 4 C Nov/Dec 2011

14. Prove by induction on n that $\sum_{i=0}^{n} i = (n(n+1))/2$ A May/June 2012

15. Construct a DFA accepting binary strings such that the third symbol from the right end is 1 C May/June 2012

16. Construct NFA without ε transitions for the NFA given below C



17. Construct an NFA accepting binary strings with two consecutive 0's. C May/June 2012

18. Explain different forms of proof with examples. U May/June 2012

19. i. Prove that if n is a positive integer such that n mod 4 is 2 or 3 then n is not a perfect square. A May/June 2012

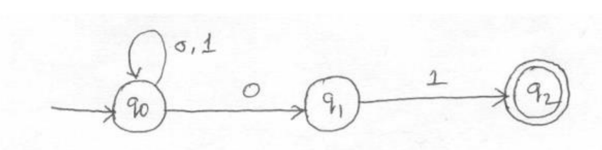
ii. Construct a DFA that accept the following language. C

{$x \in \{a, b\}$: $|x|_a$=odd and $|x|_b$= even.

20. i. Construct DFA to accept the language L= { w / w I of even length and begins with 11} C

ii. Write a note on NFA and compare with DFA. AN May/June 2013

21. Construct DFA equivalent to the NFA given below: C Nov/Dec 2013



22. Give FA accepting the following language over the alphabet C

i. Number of 1's is a multiples of 3
ii. Number of 1's is not a multiples of 3 Nov/Dec 2013

23. Discuss the application of FA. U Nov/Dec 2013

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

25. i. 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. C May/June 2014

ii. Construct NFA with ε moves which accepts a language consisting the stings of any number of a's, followed by any number of b's, followed by any number of c's. C

26. Prove that L={0^{i^2} / i is an integer; i>0} is not regular. A Nov/Dec 2014, 2015

27. i. Prove that every tree has 'e' edges and 'e+1' nodes. A Nov/Dec 2014

ii. Prove that for every integer $n \geq 0$ the number $4^{2n+1}+3^{n+2}$ is a multiple of 13. A

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

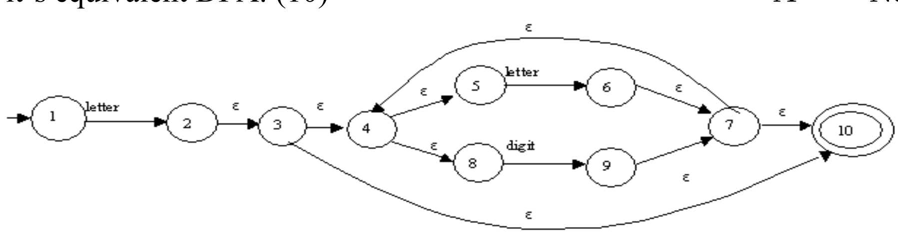
δ	0	1
a	{b,d}	{b}
b	c	{b,c}
c	d	a
d	-	a

29. Design a DFA accepts the following strings over the alphabets {0, 1} that contain a pattern 11. Prove this using mathematical induction. C April/May 2015

30. Design a NFA accept the following strings over the alphabets {0,1} that begins with 01 and ends with 11. Check for the validity of 01111 and 0110 strings. C April/May 2015

31. Prove that "A language L is accepted by some DFA if and only if L is accepted by some NFA". A

32. Consider the following ε-NFA for an identifier. Consider the ε-closure of each state and find it's equivalent DFA. (10) A Nov/Dec 2015



33. Construct a NFA that accepts all strings hat end in 01. Give its transition table and extend transition function for the input string 00101. Also construct a DFA for the above NFA using subset construction method. C May-June2016

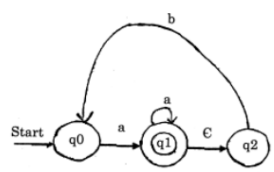
34. Construct DFA which recognize L={$b^m a b^n$/ m,n>0} C Nov-Dec2016

35. Determine the DFA from a given NFA A Nov-Dec2016

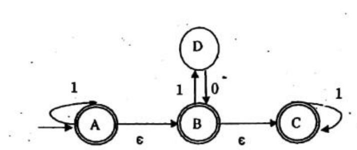
δ	a	b
q_0	$\{q_0, q_1\}$	$\{q_1\}$
q_1	ϕ	$\{q_0, q_1\}$

36. Prove for the every n>=1 by mathematical induction $\sum^{n} i3$={n(n+1)/2}^{2} E May-June 2017

37. Convert the epsilon NFA and list the difference between NFA and DFA. A Nov-Dec2017



38. Convert the following E-NFA to NFA and then convert the resultant NFA to DFA. A Nov-Dec 2018



39. Prove that a language L is accepted by some NDFA if and only if L is accepted by some DFA. E Nov-Dec 2018

40. Prove by induction on n>=1 that $\sum_{i=1}^{n} \frac{1}{i(i+1)} = \frac{n}{n+1}$. E Nov-Dec 2019.

41. Convert to a DFA, the following NFA A Nov-Dec 2019

	0	1
P(start)	{p,q}	{p}
Q	{r}	{r}
R	{ s }	-
S	{s}	{s}

42. Give NFA accepting the set of strings in (0+1)* such that two 0's are separated by a string whose length is 4i, for some i>=0. U Nov-Dec 2019

43. Construct a minimized DFA from R.E $(x+y)x(x+y)^*$. Trace for a string w=xxyx.

C Nov/Dec 2011

UNIT II REGULAR EXPRESSIONS AND LANGUAGES

Regular expression – Regular Languages- Equivalence of Finite Automata and regular expressions – Proving languages to be not regular (Pumping Lemma) – Closure properties of regular languages.

PART A

1. **Differentiate regular expression and regular language AN Nov/Dec 2012**

(Or)

What is regular expression? May/June 2013

Regular Expression	Regular Language
A regular expression is a string that describes the whole set of strings according to certain syntax rules. These expressions are used by many text editors and utilities to search bodies of text for certain patterns etc. Definition is: Let ∑ be an alphabet. The regular expression over ∑ and the sets they denote are: i. φ is a r.e and denotes empty set. ii. ε is a r.e and denotes the set {ε} iii. For each 'a' in ∑ , a^+ is a r.e and denotes the set {a}. iv. If 'r' and's' are r.e denoting the languages R and S respectively then (r+s), (rs) and (r*) are r.e that denote the sets RUS, RS and R* respectively.	A language is regular if it is accepted by some finite automaton.

2. **Give the regular expression for set of all strings ending in 00. C Nov/Dec 2010**

R.E= $(0+1)^*00$

3. **State pumping lemma for regular language. R Nov/Dec 2022 Nov/Dec 2010, 2013 , 2014 & 2017 , May-June 2016, Nov/Dec 2018,2019**

Let L be regular language then there exist a constant n (Number of states that accept the language L) such that if W is the word or set of input string in the language L then,

1. Z = UVW
2. |UV| ≤ n
3. |V| >= 1
4. UViW Є L For all i ≥ 0
5.

4. **Give the regular expression for the following C Nov/Dec 2012**

L1= set of all strings of 0 and 1 ending in 00

L2= set of all string 0 and 1 beginning with 0 and ending with 1

R1= (0+1)*00

R2=0(0+1)*1

5. Name any four CFG. U May/June 2013& May/June 2014 Nov-Dec 2016

- Union of two regular language is regular.
- Concatenation of regular language is regular.
- Closure of regular language is regular.
- Complement of regular language is regular.
- Intersection of regular language is regular.
- Difference of regular language is regular.
- Reversal of regular language is regular.
- Homomorphism of regular language is regular.
- Inverse Homomorphism of regular language is regular.

6. Is regular set is closed under complement? Justify. U May/June 2012

If L is a regular language over alphabet Σ then $\overline{L} = \Sigma^* \setminus L$ is also regular.

Proof: Let L be recognized by a DFA

$$A = (Q, \Sigma, \delta, q_0, F).$$

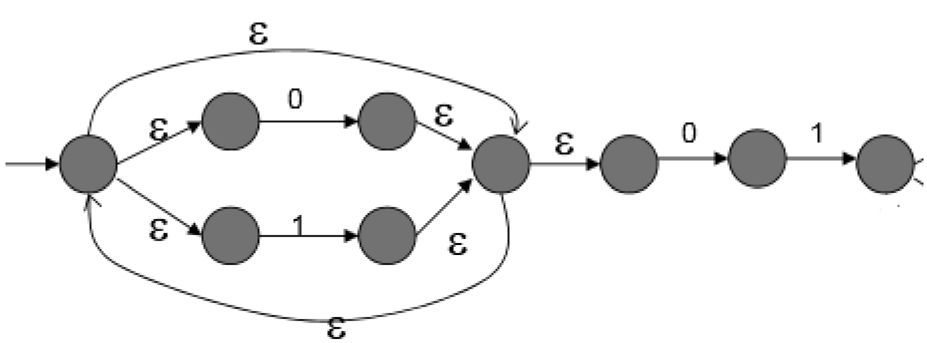
Then $\overline{L} = L(B)$ where B is the DFA

$$B = (Q, \Sigma, \delta, q_0, Q \setminus F).$$

That is, B is exactly like A, but the accepting states of A have become the nonaccepting states of B and vice-versa.

Then w is in $L(B)$ iff $\hat{\delta}(q_0, w)$ is in $Q \setminus F$, which occurs iff w is not in $L(A)$.

7. Construct NFA for the regular expression (0+1)01 C Nov/Dec 2013



8. Prove or disprove that (r+s)*=r*+s*. A Nov/Dec 2014

Replace r by *{a}* and s by *{b}*. The left side becomes all strings of *a*'s and *b*'s (mixed), while the right side consists only of strings of *a*'s (alone) and strings of *b*'s (alone). A string like *ab* is in the language of the left side but not the right.

9. Give English description of the following language (0+10)*1*. C April/May 2015

Set of all strings of 0’s and 1’s including ξ

10. Write RE for the set of strings over {0,1} that have atleast one. C Nov/Dec 2015

(0+1)*1(0+1)*

11. Show whether a language L=(0^n1^{2n}/n>0} is regular or not using pumping Lemma. E May-June 2017

Suppose L is regular. We then have some $p>0$ and some $|m|>p$
$a^p b^{2p}$
$m=uvw$
and $|uv|\leq p$ and $uv^iw\in L$ for all $i>0$
As $|uv|\leq p$
then it follows that $v=a^l$. However, as $uviw\equiv a^{p+l}b^{2p}$ it shows that as $p+l\neq 2p$ therefore L is not regular.

PART-B

1. State and explain the conversion of DFA into R.E using Arden’s theorem. Illustrate with an example. A Nov/Dec 2011

2. i. Define regular expression. R Nov/Dec 2011

ii. Show that (1+00*1)+(1+00*1)(0+10*1)*(0+10*1)=0*1(0+10*1)* A

3. Obtain minimized finite automata for the R.E (b/a)*baa. A May/June 2012s

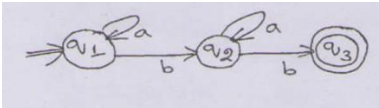
4. Prove that there exists an NFA with €- transition that accepts the regular expression r. A May/June 2012

5. Which of the following language is regular? Justify. U May/June 2012

i. L={ a^nb^m/n,m>0}

ii. L={ a^nb^n/n,>0}

6. Obtain the regular expression for the finite automata. A May/June 2012



7. i. Using pumping lemma for the regular sets, prove that the language L={$a^m b^n$/m>n} is not regular.

ii. Prove any two closure properties of regular languages. Nov/Dec 2012

8. Construct a minimized DFA from R.E 0*(01)(0/111)*. C Nov/Dec 2012

9. Discuss on the relation between DFA and minimal DFA U May/June 2013

10. i. Discuss on regular expression. U May/June 2013

ii. Discuss in detail about the closure properties of regular languages. U

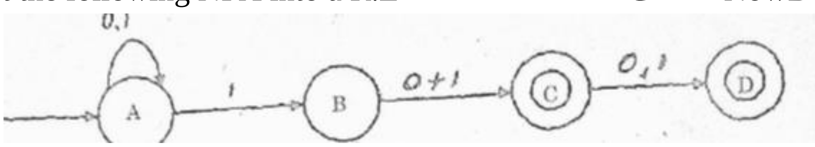
11. Prove that the following languages are not regular A May/June 2013

i. {0^{2n}/n>0}

ii. {$a^m b^n a^{m+n}$/m>0and n>0} Nov/Dec 2013

12. Discuss on equivalence and minimization of automata. U May/June 2013

13. Convert the following NFA into a R.E C Nov/Dec 2013



14. i. Design a FA for the R.E (0+1)*(00+11)(0+1*) C May/June 2014 ii. Prove that L={0^{i^2} / i is an integer; i>0} is not regular. An

Nov/Dec 2014, 2015

15. Prove that the class of regular sets is closed under complementation. A

May/June 2014

16. Construct a minimized DFA for the RE 10+(0+11)0*1. C Nov/Dec 2014

17. Explain the DFA minimization algorithm with an example. U Nov/Dec 2014

18. Find the min- state DFA for (0+1)*10. A April/May 2015

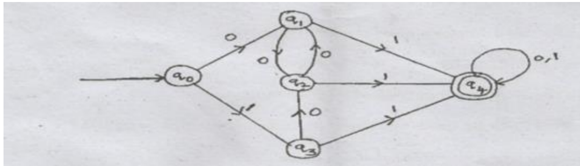
19. Find the regular expression of a language that consists of set of strings with 11 as well as ends with 00 using Rij formula. A April/May 2015

20. Construct FA equivalent to the regular expression(ab+a)* C

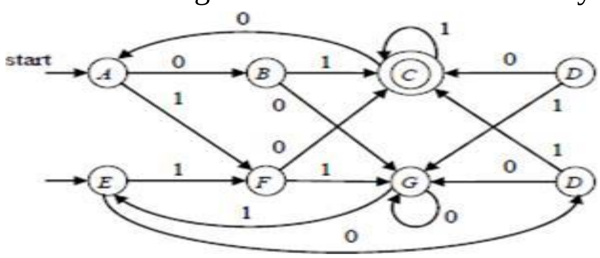
Nov/Dec 2015

21. What is Regular Expression? Write a regular expression for set of strings that consists of alternating 0's and 1's. C May-June2016

22. Minimize the FA shown in fig below and show both the given and the reduced one are equivalent. A May/June 2014



23. Write and explain the algorithm for minimization of a DFA. Using the above algorithm minimize the following DFA. A May-June2016

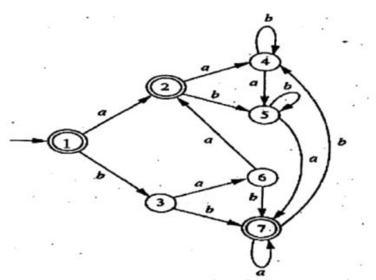


24. Construct NFA with epsilon for the R.E=(a/b)*ab and convert into DFA and further find the minimized DFA. C May-June 2017

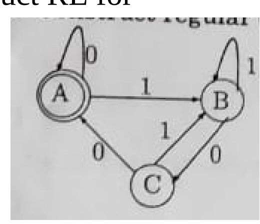
25. Show that the regular language are closed under : E Nov-Dec2017

- Union
- Intersection
- Kleen closure
- Complement
- Difference

26. Minimize the following automaton E Nov-Dec 2018



27. Construct RE for C Nov-Dec 2019



28. Prove that any language accepted by a DFA can be represented by regular expression. E Nov-Dec 2019

29. Construct a finite automata for the RE 10+(0+11)0*1 C Nov-Dec 2019

30. Prove that the following languages are not regular: E Nov-Dec 2019

i. $\{w€\{a,b\}^*/w=w^r\}$

ii. Set of string of 0's and 1's beginning with a 1, whose value treated as binary number is a prime.

UNIT III

CONTEXT FREE GRAMMAR AND PUSH DOWN AUTOMATA

Types of Grammar - Chomsky's hierarchy of languages -Context-Free Grammar (CFG) and Languages – Derivations and Parse trees – Ambiguity in grammars and languages – Push Down Automata (PDA): Definition – Moves - Instantaneous descriptions -Languages of pushdown automata – Equivalence of pushdown automata and CFG-CFG to PDA-PDA to CFG – Deterministic Pushdown Automata

1. Define CFG .Give an example. R Nov-Dec 2016

This is the way of describing language by recursive rules called production. It consists of set of variables, set of terminal symbols, and a starting variable as well as the production. G = (V,T,P,S) Where V = variables, T = Terminals, P = productions, S = starting variable.

Eg 1: $G = (V,T,P,S)$ $V = \{E\}$ $T = \{+,*,id\}$ $S = \{E\}$

E => E+E E => E*E E => id

2. What is CFL? R May/June 2013

If grammar G = (V,T,P,S) be a context free grammar then the language L(G) is a set of terminal strings that have derivation from the starting symbol.

$$\boxed{L(G) = \{ w \text{ in } T \,/\, S \overset{*}{\underset{G}{\Rightarrow}} w \}}$$

- The language generated by the CFG is called Context Free Language.

Ex: Find L(G) for the following grammar.

a) $\boxed{S \Rightarrow aSb \,/\, ab}$

S=> aSb

=> aaSbb ∵ S=>aSb

=> aaaSbbb

=> aaaaSbbbb ∵ S=>ab

∴ $S \overset{*}{\underset{G}{\Rightarrow}}$ aaaaaSbbbbb ∴ L(G) = { $a^n b^n$ / n > 1}

3. What is derivation? **R**

It is defined as $\alpha \overset{*}{\underset{G}{\Rightarrow}} \beta$ where β is derived from the symbol 'α' with the grammar 'G'.

Here, we use the production from head to body (i.e.) from starting root node expanding until it reaches the given input string.

(i.e.) R.N => w

Eg: $\boxed{\text{w=01C10}}$

4. What are the 2 types of derivation? **R**

Left most derivation:

If at each step in derivation, a production is applied to the left most variable (or) left most non-terminal then the derivation method is called left most derivation.

Eg:

$\boxed{\text{w = id+id*id}}$

E

E => E*E

E => id

E => E+E

E => id+E $\because$ E=>id

E => id+E*E $\because$ E=>E*E

E => id+id*E $\because$ E=>id

$\boxed{E \overset{*}{\underset{G}{\Rightarrow}} id+id*id}$

Right most derivation:

A derivation in which the right most variable is replaced at each step then, the derivation method is called right most derivation.

Eg:

E => E+E

E => E+E*E $\because$ E=>E*E

E => E+E*id $\because$ E=>id

E => E+id*id

$\boxed{\therefore E \overset{*}{\underset{G}{\Rightarrow}} id+id*id}$

5. What is parse tree (or) derivation tree? **R**

Parse tree is a pictorial representation of derivation, where the interior nodes are labeled by variables (or) non-terminals and leaf nodes are labeled by terminals symbols.

Eg:

$\boxed{w = 01C10}$

E => 0E0

E => 1E1

E => C

Derivation:

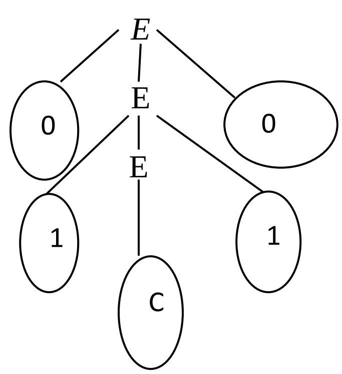
E => 0E0

=>01E10 ∵ E => 1E1

=> 01C10 ∵ E => C

$\boxed{\therefore E \overset{*}{\Rightarrow} 01C10}$

Parse tree (or) derivation tree:



6. What is ambiguous grammar? Or When do you say grammar is ambiguous? **R**
Nov/Dec 2012,2019, May/June 2013 , May/June 2014 & Nov/Dec 2022

A grammar that produces more than one parse tree (or) derivation tree for some sentence, then the grammar is said to be an ambiguous grammar. An ambiguous grammar produces more than one left most derivation (or) more than 1 RMD then, the given grammar is set to be an ambiguous grammar.

7. For the grammar defined by the productions recognize the string 1001 **and also construct the parse tree.** **A**

S => A,B
A =>0A/ξ
B => 0B/1B/ξ

8. **Consider the alphabet Σ = { a,b,(,),+,*,-, . ,ξ }. Construct a CFG that generate all the strings in Σ* that are regular expression on the alphabet, Σ. C**

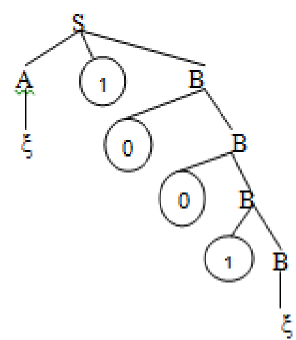
Nov/Dec 2007

E => E+E
E => E*E
E => (E)
E => E.E
E => E-E
E => a / b / ξ

w => 1001

S => A1
=>A10B ∵ B => 0B
=> A100B
=> A1001B ∵ B => 1B
=> ξ1001B ∵ A => ξ
=> 1001ξ ∵ B => ξ

PARSE TREE:



9. **Find LMD & RMD, parse tree for the following grammar. A May/June 2007**

w = 00110101

S => 0B / 1A
A => 0/0S/1AA
B => 1/1S/0BB

R.M.D:

S => 0B

=>0BB ∵ B => 0BB

=> 00B1 ∵ B => 1

=> 001S1 ∵ B => 1S

=> 0011A1 ∵ B => 1A

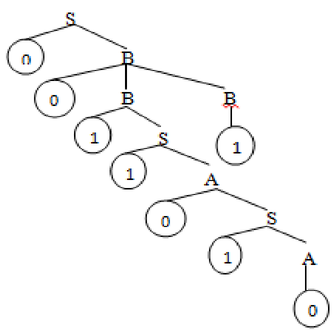
=> 00110S1 ∵ A => 0S

=> 001101A1 ∵ S => 1A

=> 00110101 ∵ A => 0

∴ $S \overset{*}{\underset{rmd}{\Rightarrow}} 00110101$

PARSE TREE:



L.M.D:

S => 0B

=> 0BB ∵ B => 0BB

=> 001B ∵ B =>1

=> 0011S ∵ B => 1S

=> 00110B ∵ S => 0B

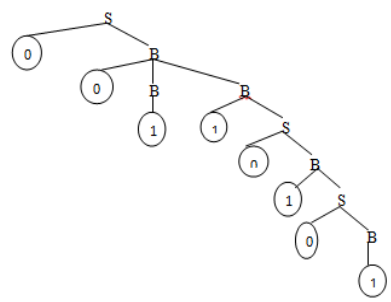
=> 001101S ∵ B => 1S

=> 0011010B ∵ S => 0B

=> 00110101 ∵ B => 1

∴ $S \overset{*}{\underset{lmd}{\Rightarrow}} 00110101$

PARSE TREE:



10. Define sentential form R

The string's are derived from the starting non-terminal is called sentential form.

If grammar G=(V,T,P,S) is a context free grammar, then α in (VUT)* such that non-terminal δ derives α is a sentential form.

$S \overset{*}{\underset{rmd}{\Rightarrow}} \alpha$ then α is left sentential form. $S \overset{*}{\underset{lmd}{\Rightarrow}} \alpha$ then α is right sentential form.

11. Let G = ({S,C}, {a,b}, P,S} where P consists of S → aCa, C→ aCa, Find L(G))? A

Solution: S → aCa

→ aaCaa C→ aCa . . .

→ a^nCa^n

→ a^nba^n C → b

L (G) = { a^nba^n ; n >0}

12. Write a grammar to recognize all prefix expressions involving all binary arithmetic operators. Construct the parse tree for the sentence "-*+abc/de" from your grammar.

A Nov/Dec 2006

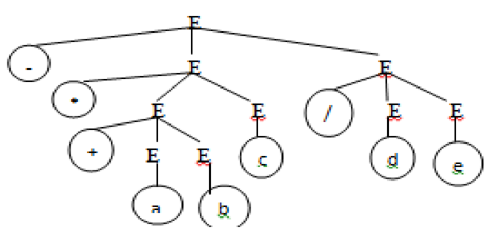
E => -EE
E => *EE
E => +EE
E => /EE
E => a / b / c / d / e

E => -EE

=>-*EEE	∵ E => *EE
=> -*+EEEE	∵ E => +EE
=> -*+aEEE	∵ E => a
=> -*+abEE	∵ E => b
=> -*+abcE	∵ E => c
=> -*+abc/EE	∵ E => /EE
=> -*+abc/dE	∵ E => d
=> -*+abc/de	∵ E => e

∴ E => -*+abc/de

PARSE TREE:



13. Write the CFG for the following CFL L(G) = {$a^m b^n c^p$/ m+n=p, m&n>1}

C **Nov/Dec 2006**

E => aEc / bTc / a /bc

T => bTc / bc

E => aEc

=>aaEcc	∵ E => aEc
=> aabTccc	∵ E => bTc
=> aabbcccc	∵ T => bc ∴

$E \underset{LMD}{\overset{*}{\Rightarrow}}$ aabbcccc

14. Let G = ({S,C}, {a,b}, P,S} where P consists of S → aCa, C→ aCa, Find L(G))? A

Solution: S → aCa

→ aaCaa   C→ aCa . . .

→ a^nCa^n

→ a^nba^n C → b

L (G) = { a^nba^n ; n >0}

15. What is the language generated by the grammar G=(V,T,P,S) where P={S->aSb, S->ab}? A

S=> aSb=>aaSbb=>…………………………..=>anbn

Thus the language L(G)={anbn | n>=1}.The language has strings with equal number of a's and b's.

16. If S->aSb | aAb , A->bAa , A->ba .Find out the CFL A

soln. S->aAb=>abab

S->aSb=>a aAb b =>a a ba b b(sub S->aAb)

S->aSb =>a aSb b =>a a aAb b=>a a a ba b bb

Thus L={anbmambn, where n,m>=1}

17. What are the properties of the CFL generated by a CFG? R

_ Each variable and each terminal of G appears in the derivation of some word in L

_ There are no productions of the form A->B where A and B are variables.

18. Find the grammar for the language L={a^{2n} bc ,where n>1 } A

let G=({S,A,B}, {a,b,c} ,P , {S}) where P:

S->Abc

A->aaA | €

19. Find the language generated by :S->0S1 | 0A | 0 |1B | 1 A

A->0A | 0 , B->1B | 1

The minimum string is S-> 0 | 1

S->0S1=>001

S->0S1=>011

S->0S1=>00S11=>000S111=>0000A111=>00000111

Thus L={ 0n 1 m | m not equal to n, and n,m >=1}

20. Construct the grammar for the language L={ a^n b a^n | n>=1}. C

The grammar has the production P as:

S->aAa

A->aAa | b

The grammar is thus : G=({S,A} ,{a,b} ,P,S)

21. Construct a grammar for the language L which has all the strings which are all palindrome over _={a, b}. C May/June 2014 , Nov/Dec 2015

G=({S}, {a,b} , P, S)
P:{ S -> aSa ,
S-> b S b,
S-> a,
S->b,
S->€ } which is in palindrome.

22. Differentiate sentences Vs sentential forms. AN

A sentence is a string of terminal symbols.

A sentential form is a string containing a mix of variables and terminal symbols or all variables. This is an intermediate form in doing a derivation.

23. What is a formal language? R

Language is a set of valid strings from some alphabet. The set may be empty, finite or infinite. L(M) is the language defined by machine M and L(G) is the language defined by Context free grammar. The two notations for specifying formal languages are:

Grammar or regular expression(Generative approach)
Automaton(Recognition approach)

24. What is Backus-Naur Form(BNF)? R

Computer scientists describes the programming languages by a notation called Backus- Naur Form. This is a context free grammar notation with minor changes in format and some shorthand.

25. Give the general forms of CNF. (Or) State CNF. R Nov/Dec 2014, Nov-Dec 2016

Every CFL is generated by a CFG in which all productions are of the form

A->BC
(or) A->a
Where A,B,C – variables
a – terminals

This form of CFG is called as Chomsky Normal Form

In order to find CNF,we need to perform the following operations.

1.Eliminate useless symbols i.e.,symbols or terminals which do not appear in any derivation of a terminal string from start symbol.
2.Eliminate €-productions which are of the form A->€ form some variable A.
3.Eliminate unit production which are of the form A->B for variables A and B.

26. Construct the CFG for the language. L(G) = $\{0^1 1^n$ / n > 1} C Nov/Dec 2013

S => 0S1/ ξ

27. What is meant by GNF? R May/June 2013

Every CFG L without ξ can be generated by a grammar for which every production is of the form A→aα, where AЄV aЄT, α is a string of variables.

28. Is the grammar ambiguous S→SS/(S)/S(S)S/ ξ? U Nov/Dec 2011

Yes. This grammar has more than one LMD and RMD.

29. Convert the following grammar into an equivalent one with no unit productions and no useless symbols S→ ABA A→ aAA/aBC/bB B→A/bB/Cb C→CC/cC. A Nov/Dec 2011

30. Generate CFG for (011+1)* A April/May2015

S→AB/BA/ ξ

A→1

B→011

31. Construct a parse tree of (a+b)*c for the grammar E→E+E/E*E/(E)/id C April/May2015

32. What do you mean by null production and unit production? Give an example. R May-June 2016

A production which is of the form A□ξ is called ξ-production.

A unit production is a production which is of form A□B, where both A and B are variables.

33. Construct a CFG fro set of strings that contain equal number of a's and b's over ∑= {a,b}. C May- June 2016

S → aSbS | bSaS | ε

34. Give language of regular expression a(b+c)*. C May- June 2017

L={w € {a,b,c}/ w starts with a and followed by any strings of b and c.

35. Generate CFG for a signed integer constant in C language. C May- June 2017

number -> sign digits

sign -> + | –

digits ->digit / digit digits

digit -> 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9

36. Derive the string "aabaab" for the following CFG E Nov/Dec 2017

$S \rightarrow aSX/b$

$X \rightarrow Xb/a$

$S \rightarrow aSX \rightarrow aaSXX \rightarrow aabXX \rightarrow aabaX \rightarrow aabaXb \rightarrow aabaab$

37. Define PDA. R May-June2016,Nov-Dec 2019

A PDA is a finite automaton with extra resource called stack.

$P = (Q, \sum, \Gamma, \delta, q_0, z_0, F)$

Where Q – Finite set of states.

$\sum$ - Finte set of input symbols.

Γ - Finite set of stack symbols.

δ - Transition States.

q_0 – Initial state.

z_0 - Initial stack symbol.

F – Final state.

38. Draw PDA accepting the language L= { $a^n c b^n$/n>0} Nov/Dec 2022

39. Construct a PDA that accepts the language generated by the grammer

$S \rightarrow aSbb$

$S \rightarrow aab$ AN

Solution:

The PDA is given by

$A = (\{q\},\{a,b\},\{S,A,B,Z,a,b\},\delta,q,S\}$

Where δ is given by

$\delta(q,z,S)=\{(q,aABB)\}$

$\delta(q,z,A)=\{(q,aB),\{q,a\}\}$

$\delta(q,z,B)=\{(q,bA),(a,b)\}$

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

40. What are the different ways of language acceptance by a PDA and define them?
. AN **Nov/Dec 2012 ,April/May2015, Nov/Dec 2015**

There are 2 ways of language acceptance

(i) Accepatnce by final state

$L(M) = \{ w \mid (q_0,w,z_0) \vdash^* (p,\varepsilon,\gamma)$ for some P in F and γ in $\Gamma^*\}$

(ii) Acceptance by empty stack

$N(M) = \{w \mid (q_0,w,z_0) \vdash^* (p,\varepsilon,\varepsilon)$ for some P in Q}

41. Define the language accepted by final state in PDA. **R**

Let $P =(Q,\Sigma,\Gamma,\delta,q_0,z_0,F)$ be a PDA.Then L(P),tha language accepted by P by final state is $L(P) =\{w \mid (q_0,w,z_0) \vdash^*(q,\varepsilon,\alpha)\}$ for some state q in F and any stack string α.

42. How do you covert CFG to PDA. **AN**

Let G =({q},T,V U T,δ,q,S) be a CFG.Then Construct a PDA P that accepts L(G) by empty as follows:

P = ({q},T ,V U T ,δ , q, S)

where δ is given by

1.For each variable A,

$\delta(q,\varepsilon,A) =\{(q,\beta) \mid A \rightarrow \beta$ is a production of P}

2.For each terminal a,

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

43. Define Deterministic PDA. **R** **Nov-Dec 2016**

A PDA $P = (Q,\Sigma,\Gamma,\delta,q_0,z_0,F)$ to be deterministic if and only if

(i) $\delta(q,a,X)$ has at most one member of any q in Q ,a in Σ or $a = \varepsilon$ and X in Γ.

(ii)If $\delta(q,a,X)$ is not empty,For some a in Σ ,then $\delta(q,\varepsilon,X)$must be empty.

44. Is it true that non – deterministic PDA is more powerful that deterministic PDA?Justify? AN

No ,NPDA is not more powerful than DPDA .Because,NPDA may produce ambiguous grammer by reaching its final state or by emptying its stack.But DPDA produces only unambiguous grammer.

45. What is the additional feature PDA has when compared with NFA?Is PDA superior over NFA in the sense of language acceptance? justify? (Or)
Compare NFA & PDA. AN Nov/Dec 2013

PDA is superior to NFA by having the following additional features.

- Stack which is used to store the necessary tape symbols and use the state to remember the conditions.
- Two ways of language acceptances, one by reaching its final state and another by emptying its stack.

46. Does a Push down Automata have memory? Justify. AN May-June2016

Yes. A pushdown automaton (PDA) is a finite automaton equipped with a stack-based memory.

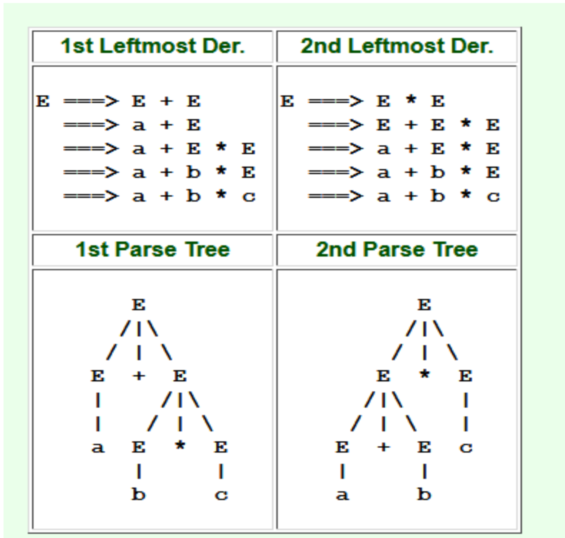
47. What are the conventional notations of PDA? R Nov-Dec2016

Transition diagram
Instantaneous description (ID)

48. Construct a RMD of (a+b)*c using the grammar and also state that whether a given grammar is ambiguous or not. C May- June 2017

$$\mathbf{E \rightarrow E + E}$$
$$\mathbf{E \rightarrow E * E}$$
$$\mathbf{E \rightarrow (E)}$$
$$\mathbf{E \rightarrow a \mid b \mid c}$$

E ===> E * E ===> E * c ===> (E) * c ===> (E + E) * c ===> (E + b) * c ===> (a + b) * c.



1st Leftmost Der.	2nd Leftmost Der.
E ===> E + E ===> a + E ===> a + E * E ===> a + b * E ===> a + b * c	E ===> E * E ===> E + E * E ===> a + E * E ===> a + b * E ===> a + b * c

```
1st Parse Tree            2nd Parse Tree

      E                         E
     /|\                       /|\
    / | \                     / | \
   E  +  E                   E  *  E
   |    /|\                 /|\    |
   |   / | \               / | \   |
   a  E  *  E             E  +  E  c
      |     |             |     |
      b     c             a     b
```

Since this grammar has two different parse tree for the string (a+b)*c, this grammar is ambiguous grammar.

49. Differentiate PDA acceptance by empty stack with acceptance by final state. A May- June 2017

Final State Acceptability

In final state acceptability, a PDA accepts a string when, after reading the entire string, the PDA is in a final state. From the starting state, we can make moves that end up in a final state with any stack values. The stack values are irrelevant as long as we end up in a final state.

For a PDA (Q, ∑, S, δ, q_0, I, F), the language accepted by the set of final states F is −

L(PDA) = {w | (q_0, w, I) ⊢* (q, ε, x), q ∈ F} for any input stack string x.

Empty Stack Acceptability

Here a PDA accepts a string when, after reading the entire string, the PDA has emptied its stack.

For a PDA (Q, ∑, S, δ, q_0, I, F), the language accepted by the empty stack is −

L(PDA) = {w | (q_0, w, I) ⊢* (q, ε, ε), q ∈ Q}

50. What is an instantaneous description of PDA? -R Nov-Dec 2018

The instantaneous description (ID) of a PDA is represented by a triplet (q, w, s) where

- **q** is the state
- **w** is unconsumed input
- **s** is the stack contents

51. When pushdown automata is said to be deterministic?

If, in every situation, at most one such transition action is possible, then the automaton is called a deterministic pushdown automaton (DPDA). In general, if several actions are possible, then the automaton is called a general, or nondeterministic, PDA.

PART-B

1. What is deterministic PDA? Explain with an example. U Nov/Dec 2010
2. Is NPDA and DPDA equivalent? Illustrate with an example. U Nov/Dec 2011
3. (i) Construct the PDA for the Language L= {WCWR | W is in (0+1)*. C
   Nov/Dec 2010, May/June 2012, Nov/Dec 2013
   (ii) Let L is a context free language. Prove that there exists a PDA that accepts L. A

4. Construct the PDA accepting the language { $(ab)^n$/n>0} by empty stack. C
   Nov/Dec 2012

5. a. Construct a transition table for PDA which accepts the language L={ $(a^{2n}b^n$/n>0}
   Trace your PDA for the input with n=3. C Nov/Dec 2012
   b. Find the PDA equivalent to the give CFG with the following productions.
   S→A A→BC B→ba C→ac

6. Construct PDA for the Language L= {WW^R | W is in (a+b)*}. C
   May/June 2013 & 2016

7. Construct the PDA accepting the language L= { a^nb^n/n>0} by empty stack and final state.
   C May/June 2014
8. Convert the grammar S→0S1/A: A→1A0/S/ ε into PDA that accepts the same language by empty stack. Check whether 0101 belongs to N(M). A May/June 2014
9. 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.
   A Nov/Dec 2014,2019
10. What are the different types of language acceptances by a PDA and define them. Is it true that the language accepted by PDA by these different types provides different languages?
    U Nov/Dec 2011
11. Convert the grammar S→aSb/A, A→bSa/S/ε to PDA that accepts the same language by empty stack. A Nov/Dec 2011
12. Discuss the equivalence between PDA and CFG.
    May/June 2012, May/June 2013, Nov/Dec 2013, May/June 2014

13. Construct a PDA for the language L={x€{a,b}*/ $n_a(x)>n_b(x)$} C April/May 2015

14. Convert the following CFG to a PDA. A Nov/Dec 2015
    S→aAA, A→aS|bS|a

15.Design a PDA to accept {0n1n|n>1}. Draw the transition diagram for the PDA. Show by instantaneous description that the PDA accepts the strings'0011'. C Nov/Dec 2015

16.Convert PDA to CFG.PDA is given by P=({p,q},{0,1},{X,Z}, δ,q,Z), δ is defined 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)}. A Nov/Dec 2015

17. What are deterministic PDA's? Give example for non-deterministic and deterministic PDA.
    U Nov/Dec 2015

18. What is an instantaneous description that the PDA? How will you represent it? Also give three important principles of ID and their transactions. U May-June2016

19. Explain acceptance by final state and acceptance by empty stack of a Push down Automata.
    U May-June2016

20. Outline an ID of a PDA. U Nov-Dec2016

21. With an example, explain the procedure to obtain a PDA from the given CFG. U Nov-Dec2016

22. Construct a DPDA for even length palindrome. C May- June 2017

23. Prove "If PDA P is constructed from CFG G by the above construction, then N(P)=L(G)". E May- June 2017

24. Convert the CFG to PDA and Verify for (a+b) and a++ A May- June 2017

 $I \rightarrow a/b/Ia/Ib/I0/I1$

 $E \rightarrow I/E+E/E*E/(E)$

25. Find PDA that accept the given CFG E May- June 2017

 $S \rightarrow XaaX$

 $X \rightarrow aX/bX/\varepsilon$

26. Construct PDA for the language $a^n b^m a^{n+m}$. CMay- June 2017

27. Prove that deterministic and non-deterministic PDA are not equivalent. E May- June 2017

28. a. Write a grammar G to recognize all prefix expressions involving all binary arithmetic operator. Construct a parse tree for the sentence '-*+abc/de' using G. C

 b. Show that the grammar G is ambiguous $S \rightarrow SbS/a$ A May/June 2014

 c. Construct a CFG for $\{0^m 1^n / 1 \leq m \leq n\}$ C Nov/Dec 2014

29. Explain about parse tree. For the following grammar. U May/June 2013

$S \rightarrow aB|bA$

$A \rightarrow a|aS|bAA$

$B \rightarrow b|bS|aBB$

For the string aaabbabbba, Find i. LMD ii. RMD iii. Parse tree Nov/Dec 2015

30. a. Is the grammar $E \rightarrow E+E/E*E/id$ is ambiguous? Justify your answer. AN

 b. Find the CFL for the following grammars A May/June 2012

(1) $S \rightarrow asbs / bsas / \varepsilon$

(2) $S \rightarrow asb / ab$

31. If $S \rightarrow aSb/aAb$, $A \rightarrow bAa/ba$ is CFG. Determine CFL AN Nov/Dec 2011

32.Let G=(V,T,P,S) be a CFG 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. A Nov/Dec 2015

33. 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…..9. Find a parse tree for the string 1+2*3 A

34. What is ambiguous grammar? Explain with an example.

U Nov/Dec2015, May-June2016

35. Show the derivation steps and construct derivation tree for the string ababbb' by using left most derivation with the grammar. A May-June2016

S —> AB / ξ, A—> aB, B—> Sb

36. Construct a CFG for the regular expression (011+1) (01). C May-June2016

37. Construct CFG for the language L={a^n/n is odd} C Nov-Dec2016

38. Define derivation tree. Explain its uses with an example. U Nov-Dec2016

39. Construct a CFG to generate even and odd set of palindrome over alphabet {a,b}.

C Nov-Dec2017

40. Generate CFG for the language L={$0^i 1^j 0^k$/j>i+k} C Nov-Dec2017

41. Show that the following grammar is ambiguous: S→SbS/a. An Nov-Dec 2018

42. Convert the CFG to PDA : S→aS/bS/a/b A Nov-Dec 2018

43. What is DPDA? Comment on the language accepting capabilities of DPDA.

U Nov-Dec 2018

44. Give the regular expression of the language generated by the CFG given below:
S→ aS/bS/a/b. Convert the RE to E-NFA

45. Convert the PDA to CFG A Nov-Dec 2018

$M = (\{q_0, q_1\}, \{0,1\}, \{X, Z_0\}, \delta, q_0, Z_0, \Phi)$ and δ is given by

$\delta(q_0, 0, Z_0) = \{(q_0, XZ_0)\}, \delta(q_1, 1, X) = \{(q_1, \epsilon)\},$

$\delta(q_0, 0, X) = \{(q_0, XX)\}, \delta(q_1, \epsilon, X) = \{(q_1, \epsilon)\},$

$\delta(q_0, 1, X) = \{(q_1, \epsilon)\}, \delta(q_1, \epsilon, Z_0) = \{(q_1, \epsilon)\}.$

UNIT IV

NORMAL FORMS AND TURING MACHINES

Normal forms for CFG – Simplification of CFG- Chomsky Normal Form (CNF) and Greibach Normal Form (GNF) – Pumping lemma for CFL – Closure properties of Context Free Languages –Turing Machine : Basic model – definition and representation – Instantaneous Description – Language acceptance by TM – TM as Computer of Integer functions – Programming techniques for Turing machines (subroutines).

PART A

1. **What are the three ways to simplify a context free grammar? R**

   _ By removing the useless symbols from the set of productions.

   _ By eliminating the empty productions.

   _ By eliminating the unit productions.

2. **What are the closure properties of CFG? U Nov/Dec 2017**

   Union : If L1 and If L2 are two context free languages, their union L1 ∪ L2 will also be context free.
   Concatenation : If L1 and If L2 are two context free languages, their concatenation L1.L2 will also be context free.
   Kleene Closure : If L1 is context free, its Kleene closure L1* will also be context free.
   Intersection and complementation : If L1 and If L2 are two context free languages, their intersection L1 ∩ L2 need not be context free.

3. **State the pumping lemma for CFL.R**
   May/June 2012, Nov/Dec 2012, May/June 2014, April/May2015

   Let L be a CFL then there exist a constant M such that if Z is any word in language L and $|Z| \geq n$ then we may write the above statements.

   By pumping lemma,

   $Z = UVWXY \quad |Z| >= n$
   $|VWX| \leq n$
   $|VX| >= 1$
   $UV^iWX^i Y \in L$ For all $i \geq 0$

4. **Give the steps to eliminate useless symbols.** **R** **Nov/Dec 2017**
   1. Find the non-generating variables and delete them, along with all productions involving non-generating variables.
   2. Find the non-reachable variables in the resulting grammar and delete them, along with all productions involving non-reachable variables.

5. **Show that CFLs are closed under substitutions** **A** **Nov/Dec 2014**

If L is a Context – free language over alphabet £ ,and S is a substitution on £ such that S(a) is a CFL for each a in £ ,then S(L) is a CFL.

Proof:

The idea here is that for a CFG,replce each terminal a by the start symbol for language S(a).The result is a single CFG that generates S(L).

Let G =(V,£,P,S) be a grammer for L.

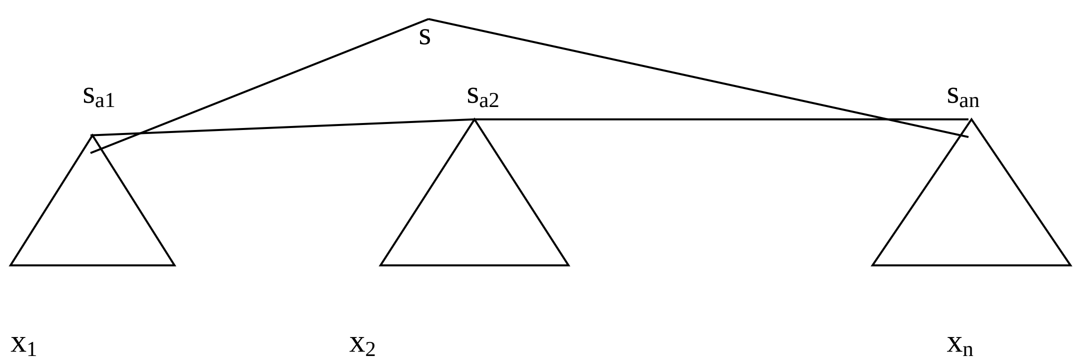
and G_a = (V_a , T_a , P_a ,S_a) be a grammer for each a in £.

Construct a new grammer G^1 = (V^1,T^1,P^1,S) for S(L).

Where

- V^1 is the union of V and V_a .[for all a in £]
- T^1 is the union of all T_a.
- P^1 is given by
  - P_a for a in £.
  - P where each terminal a is replaced by S_a.

Thus all parse trees in grammer G_1 sart out with parse trees in G but all nodes have labels that are S_a for some a in £.Then the generation of each such node produces a parsertree of G_a whose yield belongs to S(a).



(Parse tree of G^1 due to substitution)

6. Show that L={a^p/ P is prime } is not context free. E Nov/Dec 2017

Suppose that L be a context-free language and M be its corresponding Finite Automata with m number of states. Let w be a string which belongs to context-free language L with |w|=n where n is a prime number such that $n \geq m$. Hence, w can be decomposed as w=uvwxy such that $|vwx| \leq n$ and $|vx| > 0$.

Since $w \in L$ with |w|=n and w= uvwxy therefore we can get |uvwxy| = n (prime number). Means we may say that if length of a^n is prime no then it is a regular language. Now Since w = uvwxy $\in$ L, then for L to be context-free uv^iwx^iy should also belong to L for every value of i. Then we can say that the length of $uv^iwx^iy = |uv^iwx^iy|$ should be a prime number.

$|uv^iwx^iy| = |uvwxy| + (i-1)*|vx|$

let |vx| = k where k > 0 and i = n+1 then
$|uv^iwx^iy| = |uv^iwx^iy| + (i-1)*|vx| = n + (n+1-1)*k = n+n*k = n*(1+k)$ => which is composite for i=p+1. Hence, uv^iwx^iy not belongs to L for all values of i. Therefore, L is not a context-free language.

7. List the closure properties of CFL. R May/June 2013, Nov/Dec 2013, Nov/Dec 2022

- Substitutions
- Union
- Concatenation
- Closure and Positive Closure
- Homomorphism
- Reversal
- Intersection
- Inverse Homomorphism

8. Define Turing Machine. R Nov/Dec 2010,2015& 2017 , May/June 2014 & 2016

The Turing machine is denoted by
$M=(Q,\Sigma,\vdash,\delta,q_0,B,F)$ Where Q –finite set of states
Σ -finite set of allowable tape symbols
a symbol of $\vdash$, a blank
Σ- set of input symbols
$q_0 \in Q$- start state
F- set of final state
δ-Transition function mapping
$Q \times \vdash \rightarrow Q \times \vdash \times \{L,R\}$
Where L,R –Directions

9. What are the required fields of an instantaneous description or configuration of a TM? R Nov-Dec2016

It requires

- The state of the TM
- The contents of the tape
- The position of the tape head on the tape.

10. What is multiple tracks Turing machine? R

A Turing machine in which the input tape is divided into multiple tracks where each track having different inputs is called multiple track Turing machine.

11. What is multidimensional Turing machine? R

The Turing machine which has the usual 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 new symbol and moves its tape head in one of 2K directions along one of K axes.

12. When is a function f said to be Turing computable? U

A Turing Machine defines a function y=f(x) for strings x,yЄΣ*, if

$q_0 x \vdash^* q_f y$ where q_0 – initial state ,q_f final state

A function f is 'Turing computable' if there exist a Turing machine that perform a specific function.

13. What is off line Turing machine? R

An off-line Turing machine is a multitape Tm whose input tape is read only. The Turing machine is not allowed to move the input tape head off the region between left and right end markers.

14. List out the different techniques for TM construction. R Nov/Dec 2013

1. Storage in the finite control (or) State.
2. Multiple tracks.
3. Subroutines.
4. Checking off symbols

16. What is Universal Turing machine? R Nov/Dec 2013, 2016

A universal Turing machine is a Turing machine Tu that works as follows.

It is assumed to receive an input string of the form e(T)e(z), where T is an arbitrary TM, z is a string over the input alphabet of T , and e is an encoding function whose values are strings in {0, 1}*. The computation performed by Tu on this input string satisfies these two properties:

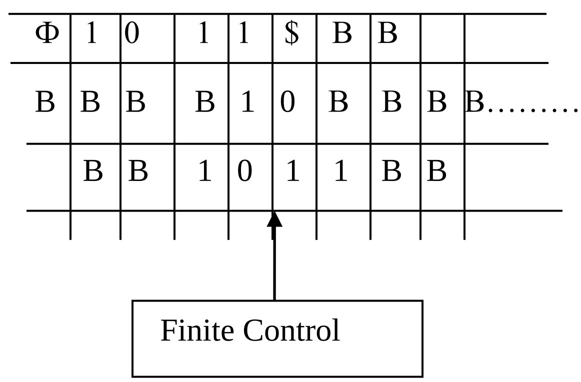
1. Tu accepts the string e(T)e(z) if and only if T accepts z.
2. If T accepts z and produces output y, then Tu produces output e(y).

17. Define multitape TM. R Nov/Dec 2014, Nov/Dec 2015

A **Multi-tape Turing machine** is like an ordinary Turing machine with several tapes. Each tape has its own head for reading and writing. Initially the input appears on tape 1, and the others start out blank.

A k-tape Turing machine can be described as a 6-tuple $M = \langle Q, \Gamma, s, b, F, \delta \rangle$ where:

- Q is a finite set of states
- Γ is a finite set of the tape alphabet
- $s \in Q$ is the initial state
- $b \in \Gamma$ is the blank symbol
- $F \subseteq Q$ is the set of final or accepting states
- $\delta : Q \times \Gamma^k \rightarrow Q \times (\Gamma \times \{L, R, S\})^k$ is a partial function called the transition function, where k is the number of tapes, L is left shift, R is right shift and S is no shift.



(A Three Track Turing Machine)

14. **List the primary objectives of TM. R Nov-Dec2016**

A Turing machine is an abstract machine that manipulates symbols on a strip of tape according to a table of rules; to be more exact, it is a mathematical model of computation that defines such a device. Despite the model's simplicity, given any computer algorithm, a Turing machine can be constructed that is capable of simulating that algorithm's logic.

15. **What are the differences between a Finite automata and a Turing machine? A**
May-June2103

Finite Automata	**Turing Machine**
Finite Automation is a 5-tuple (Q, ∑,δ,q_0,F) where Q be a finite set of states ∑ be a finite set of symbols δ be a transition function mapping from Q X ∑ to Q q_0 the initial state and F the set of final state	A Turing Machine M is a 7-Tuple M = (Q,∑,┬,δ,q_0,B,F) Where Q – finite set of states ∑ - finite set of input symbols ┬ - finite set of tape symbols. δ – Transition function mapping the states of finite automaton and tape symbols to states,tape symbols and movement of the head. i.e., Q x ┬ -> Q x ┬ x {L,R} q_0 ∑ Q is the intial state F ≤ Q is the set of final states. B ∑ ┬ is the blank symbol.

16. **What is halting problem. R** **May-June2107**

In computability theory, the halting problem is the problem of determining, from a description of an arbitrary computer program and an input, whether the program will finish running or continue to run forever.

17. **Write short note on chomskian hierarchy of languages.** **U**
May-June2107 Nov/Dec 2018

- Chomsky Hierarchy is a broad classification of the various types of grammar available
- These include Unrestricted grammar, context-free grammar, context-sensitive grammar and restricted grammar
- Grammars are classified by the form of their productions.
- Each category represents a class of languages that can be recognized by a different automaton

18. **Give the configuration of Turing Machine.** **Nov/Dec 2017**

A **configuration** for a Turing machine is an ordered pair of the current state and the tape contents with the symbol currently under the head marked with underscore. For example (q , aab**a**bb) shows that the Turing machine is currently in state q, the taper contents are the string aababb and the head is reading the last a of the string.

We write (p , x**a**y) |- (q , z**b**w) if the Turing machine goes from the first configuration to the second in one move, and (p , x**a**y) $|-^*$ (q , z**b**w) if the Turing machine goes from the first configuration to the second in zero or more moves.

19. **Differentiate multihead and multi tape Turing machine. U Nov/Dec 2018**

Multihead TM	Multi tape TM
A multi-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.	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.

20. **What are the advantages of having a normal form for a grammar? U**
Nov/Dec 2019, Nov/Dec 2022

While PDAs can be used to parse words with any **grammar**, this is often inconvenient. **Normal forms** can give us more structure to work with, resulting in easier parsing algorithms.

21. **Define the language recognized by the TM. R Nov/Dec 2019**

A TM accepts a language if it enters into a final state for any input string w. A language is recursively enumerable (generated by Type-0 grammar) if it is accepted by a Turing machine. A TM decides a language if it accepts it and enters into a rejecting state for any input not in the language.

22. **When do you say a TM is an algorithm? U Nov/Dec 2019**

If an algorithm exists, then a turing machine can run it!.”In other words, what all can be done by an algorithm can also be done by the Turing Machine.

PART B

1. Is the language $L = \{a^n b^n c^n \mid n>=1\}$ is context free? Justify. AN Nov/Dec 2010
(Or) Show that the Language $L=\{ a^i b^i c^i / i \geq 1\}$ is not context free. A May/June 2014
2. Discuss the closure properties of CFL. U Nov/Dec 2010,May/June 2012, Nov/Dec 2012, May/June 2013
3. Show that language $\{0^n1^n2^n/n>=1\}$ is not CFL. A Nov/Dec 2014& Nov/Dec 2015
4. State the pumping lemma for CFL. Use pumping lemma to show that the language L= $\{a^ib^jc^k/i<j<k\}$ is not a CFL. A May-June2016
5. State and explain the pumping Lemma for CFG. U Nov-Dec2016
6. Explain pumping Lemma for CFL. U May- June 2017
7. Convert the following grammar into GNF A Nov/Dec 2013
$S \rightarrow XY1/0, X \rightarrow 00X/Y, Y \rightarrow 1X1$

8. Construct the following grammar in CNF **C**

$A \to BCD|b$

$B \to Yc|d$

$C \to gA/c$

$D \to dB|a$

$Y \to f.$

9. Construct the following grammar in CNF C Nov/Dec 2012

$S \to cBA, S \to A, A \to cB, A \to AbbS, B \to aaa.$

10. Find GNF for the grammar A May/June 2012

$S \to AA/\perp$
$A \to SS/\theta$

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

12. Given the CFG G, find CFG G' in CNF generating the language L(G)-{^}

S→AACD

A→aAb/^

C→aC/a

D→aDa/bDb/^ A April/May 2015

13. Construct a reduced grammar equivalent to the grammar G = (N, T, P, S) where, N = {S. A, C, D, E} T= {a, b} C

P = { S —> aAa, A —> Sb, A —> bCC, A —> DaA, C —> abb. C —> DD, E→ aC, D —> aDA}. C May-June2016

14. What is the purpose of normalization? Construct the CNF and GNF for the following grammar and explain the steps. A May-June2016

$S \to aAa \mid bBb \mid \varepsilon$
$A \to C \mid a$
$B \to C \mid b$
$C \to CDE \mid \varepsilon$
$D \to A \mid B \mid ab$

15. Obtain a Grammar in CNF A Nov-Dec2016

$$S \rightarrow aAbB$$
$$A \rightarrow aA \mid a$$
$$B \rightarrow bB \mid b.$$

16. Given the CFG G, find CFG G' in CNF generaring the language L(G)- { ξ}

S→AACD C May- June 2017

A→aAb\ ξ

C→aC\a

D→aDa\bDb\ ξ

17. Convert the following grammar G into Greibach Normal Form

S→XA\BB C May- June 2017

B→b\SB

X→b, A→a

18. Find an equivalent grammar in CNF for the grammar:

S→bA/aB E Nov-Dec2017

A→bAA/aS/a B→aBB/bS/b

19. Eliminate the unit production of the following grammar

S→A/bb A Nov-Dec2017

A→B/b

B→S/a

20. Design a TM that accepts the language of odd integers written in binary. C Nov/Dec 2011

21. What are the applications of TM. R Nov/Dec 2012

22. Construct the Turing machine for the language L = $\{0^n 1^n \mid n >= 1\}$ C Nov/Dec 2010

23. i. Explain the difference between tractable and intractable problems with examples. A
ii. What is halting problem? Explain. U Nov/Dec 2010, Nov/Dec 2015

24. i. State the techniques for TM construction. Illustrate with a simple language.U
ii. Explain the different models of TM. U Nov/Dec 2011

25. State the halting problem of TMs. Prove that the halting problem of TM over {0,1}* as unsolvable. U Nov/Dec 2011

26. Explain any two higher level techniques for TM construction.U May/June 2012

27. Construct the Turing machine for the language L = $\{1^n 0^n 1^n \mid n >= 1\}$ C May/June 2012

28. a. Design TM which reverses the given string {abb}. C Nov/Dec 2012
    b. Write briefly about the programming techniques for TM. U May/June 2013
29. Explain TM as a computer of integer functions with an example. U Nov/Dec 2013
30. Write short notes on the following: U Nov/Dec 2013
    a. Two way infinite tape TM
    b. Multiple tracks TM
31. a. Design a TM to accept the language L$0^n 1^n$ /n>=1} and stimulate its action on the input 0011. C May/June 2014, Nov/Dec 2015
    b. Write shot note on checking off symbols. U
32. Design a TM, M to implement the function “multiplication” using the subroutine “copy”. C Nov/Dec 2014
33. Construct TM to perform copy operation. C April/May 2015
34. Explain the programming techniques for TM construction. U Nov/Dec 2015,2016
35. Describe the Chomsky hierarchy of languages. U Nov/Dec 2015&2017 , May- June 2017
36. Construct a Turing Machine to accept palindromes. Trace the string "abab" and "baab". C May-June2016
37. Explain the variations of Turing Machine. U May-June2016
38. Explain Halting problem. Is it solvable or unsolvable problem? Discuss.
39. Describe the Chomsky hierarchy of languages with example. What are the devices that accept these languages? U
40. Write about Multi-tape TM. U Nov-Dec2016
41. Highlight the implications of halting problems. U Nov-Dec2016
42. Construct a TM to reverse the given string. C May-June2107
43. Explain Multi tape and Multi head Turing machine with suitable example. U May-June2107
44. Construct TM that replace all occurrence of 111 by 101 from sequence of 0’s and 1’s. C Nov-Dec2017
45. Explain techniques for TM construction. U Nov-Dec2017
46. Prove that Halting problem is undecidable. E Nov-Dec2017
47. Consider two tape TM and determine whether the TM always write a nonblank symbol on its second tape during the computation on any input string ‘w’. formulate this problem as a language and show it is undecidable. E Nov-Dec2017
48. Simply the following grammar by eliminating null productions, unit production and useless symbols and then convert to CNF. E Nov-Dec2018

    S→ABC/BaB

    A→aA/ BaC/aaa

    B→bBb/a/D

    C→CA/AC

    D→ ξ

52. Convert the following grammar G into Greibach Normal Form

    S→AB, A→BS/b, B→SA/a E Nov-Dec2018

53. Prove that the language L={$a^n b^n c^n$/n>=1} is not context free using pumping lemma E Nov-Dec2018

54. Give the five tuple representation of a TM and explain the representation. Define the language accepted by a TM. U Nov-Dec2018

55. Design TM that accepts the language L={SS/ S is in {a,b}* C Nov-Dec2018

56. Design TM for $L=\{a^n b^n c^n / n>=1\}$ U Nov-Dec2018

57. Suppose L=L(G) for some CFG G=(V<T<P<S) then prove that L-{ ξ } is L(G') for CFG G' with no useless symbols or ξ- production. E Nov-Dec2019

58. State and prove GNF. E Nov-Dec2019

59. Design TM to compute proper subtraction. C Nov-Dec2019

UNIT V UNDECIDABILITY

Unsolvable Problems and Computable Functions –PCP-MPCP- Recursive and recursively enumerable languages – Properties - Universal Turing machine -Tractable and Intractable problems - P and NP completeness – Kruskal's algorithm – Travelling Salesman Problem- 3-CNF SAT problems.

PART A

1. **When a language is said to be recursively enumerable? U Nov/Dec 2010, May/June 2012, Nov/Dec 2012, May/June 2013, Nov/Dec 2013,May/June 2014, April/May2015**

   A language is recursively enumerable if there exists a Turing machine that accepts every string of the language and does not accept strings that are not in the language.

2. **Define Non Recursive language. R Nov/Dec. 2022**

   If the languageL is not recursively enumerable, then there is no algorithm for listing the members of L. It might be possible to define L by specifying some property that all its members satisfy, but that property can't be computable.

3. **When a language is said to be recursive? U**

   A language L is said to be recursive if there exists a Turing machine M that accepts L, and goes to halt state or else M rejects L.

4. **Define decidable problems. R**

   A problem is said to be decidable if there exists a Turing machine which gives one 'yes' or 'no' answer for every input in the language.

5. **Define undecidable problems. R**
   If a problem is not a recursive language, then it is called undecidable problem.

6. **Define universal language. R**
   A universal Turing machine M_u is an automaton, that given as input the description of any Turing machine M and a string w, can simulate the computation of M on w.

7. **Define problem solvable in polynomial time. R**
   A Turing machine M is said to be of time complexity T(n) if whenever m is given an input w of length n, m halts after making at most T(n) moves, regardless of whether or not m accepts.

8. **Define the class P and NP. R May/June 2013, 2014 & Nov-Dec 2019**
   P consists of all those languages or problems accepted by some Turing machine that runs in some polynomial amount of time, as function of its input length.
   NP is the class of languages or problems that are accepted by nondeterministic TM's with a polynomial bound on the time taken along any sequence of non – deterministic choices.

9. **Define NP – Complete Problem. R Nov-Dec2016**
   A language L is NP – complete if the following statements are true.
   (i)L is in NP.
   (ii)For every language L^1 in NP there is a polynomial – time reduction of L^1 to L.

10. **Write the Significance of NP-Complete Problem. R Nov-Dec2022**
    NP-complete languages are significant because all NP-complete languages are thought of having similar hardness, in that process solving one implies that others are solved as well. If some NP-complete languages are proven to be in P, then all of NPs are proven to be in P.

11. **What are tractable problems? R Nov-Dec2017**
    The problems which are solvable by polynomial – time algorithm are called tractable problems. For Eg. The complexity of the Kruskal's algorithm is 0(e(e+m)where e ,the number of edges and m,the number of nodes.

12. **What are the properties of recursively enumerable sets which are undecidable? R**
    1.Emptiness
    2.Finiteness
    3.Regularity
    4.Context – freedom.

13. What are the properties of recursive and recursively enumerable language?
R Nov-Dec2017

(i) The complement of a recursive language is recursive.

(ii) The union of two recursive languages are recursive the union of two recursively enumerable languages are recursively enumerable.

(iii)If a language L and L' are both recursively enumerable, Then L is recursive.

14. Mention the difference between decidable and undecidable problems. AN Nov/Dec 2010

Decidable Problem	**Undecidable Problem**
A problem is said to be decidable if there exists a Turing machine which gives one 'yes' or 'no' answer for every input in the language.	If a problem is not a recursive language, then it is called undecidable problem.

15. **Show that any PSPACE-hard language is also NP-hard. A Nov/Dec 2010**

16. Mention the difference between P and NP problems. AN May/June 2012

P problems	**NP problems**
P consists of all those languages or problems accepted by some Turing machine that runs in some polynomial amount of time, as function of its input length.	NP is the class of languages or problems that are accepted by nondeterministic TM's with a polynomial bound on the time taken along any sequence of non – deterministic choices.

17. When we say a problem is decidable? Give example of undecidable problem. U Nov/DEC 2012, Nov/Dec 2015

A problem is said to be decidable if there exists a Turing machine which gives one 'yes' or 'no' answer for every input in the language.
E.g Halting problem

18. Give examples for NP – Complete Problem. U Nov/Dec 2014

1. Complete sub graph problem is NP-complete.
2. The *k*-colorability problem is *NP*-complete.
3.

19. Differentiate Recursive and Non-recursive language. AN April/May2015

20. When is a Recursively Enumerable language said to be Recursive?
U May-June2016

A language is Recursively Enumerable (RE) if some Turing machine accepts it.
A TM M with alphabet _ accepts L if L = {w ∈ _∗|M halts with input w }

Let L be a RE language and M the Turing Machine that accepts it., for w ∈ L, M halts in final state. For w /∈ L, M halts in non-final state or loops forever.
A language is Recursive (R) if some Turing machine M recognizes it and halts on every input string, w ∈ _∗. Recognizable = Decidable. Or A language is recursive if there is a membership algorithm for it. Let L be a recursive language and M the Turing Machine that accepts (i.e. recognizes) it. For string w, if w ∈ L, then M halts in final state. If w /∈ L, then M halts in non-final state.

21. Identify whether 'Tower of Hanoi' problem is tractable or intractable. Justify your answer. U May-June2016

'Tower of Hanoi' problem is intractable.
Intractable Problem: a problem that cannot be solved by a polynomial-time algorithm. The lower bound is exponential.
Towers of Hanoi: we can prove that any algorithm that solves this problem must have a worst-case running time that is at least 2^n-1.

22. What is primitive recursive function? R May-June2107
Define the primitive recursion operation. R Nov-Dec 2018

Function is considered primitive recursive if it can be obtained from initial functions and through finite number of composition and recursion steps.

23. Define NP completeness. R May-June2107

A problem is NP-complete if answers can be verified quickly, and a quick algorithm to solve this problem can be used to solve all other NP problems quickly.

PART B

1. Prove that ‘If ‘L’ is a recursive language, then L’ is also a Recursive Language’. E
2. Prove that ‘If a language L and L’ are recursively enumerable (RE) , then L is Recursive’. E
3. Prove that (i) Lu is recursively enumerable but not recursive. E
   (ii) Non empty language Lne is recursively enumerable.
4. Find the languages obtained from the following operations: A
   (i) Union of two recursive languages. (6) Nov/Dec 2014
   (ii) Union of two recursively enumerable languages (6)
   (iii) L if L and complement of L are recursively enumerable (4)
5. a) Show that the following language is not decidable. E
   L={<M>| M is a TM that accepts the string aaab}. (8)
   b) Discuss the properties of Recursive and Recursive enumerable languages. U (8)
6. Prove that the universal language Lu is recursively enumerable. E
   May/June 2014 , Nov/Dec 2014, Nov/ Dec 2015
7. Define the universal language and show that it is recursively enumerable but not recursive. U
8. Whether the problem of determining given recursively enumerable language is empty or not? Is decidable? Justify your answer. AN

9. Define Universal language Lu. Show that Lu is recursively enumerable but not recursive. U
10. Explain the Halting problem. Is it decidable or undecidable problem? U Nov/DEC 2011 , Nov/Dec 2012
11. Explain the difference between tractable and intractable problems with examples. U Nov/Dec 2010
12. Write short notes on: i. Recursive and recursively enumerable language
    ii. NP hard and NP complete Problems U Nov/Dec 2011
13. Discuss the properties of recursive languages. U May/June 2012
14. Explain any two undecidable problems with respect to TM. U May/June 2012 , May/June 2013
15. Discuss the difference between NP-complete and NP-hard problems. May/June 2012 U
16. Write note on NP problems. U Nov/Dec 2012, Nov/Dec 2013
17. Explain about “A language that is not Recursively Enumerable”.U May/June 2013
18. Prove that for two recursive language L1 and L2 their union and intersection is recursive. A Nov/Dec 2013.
19. Explain post correspondence problems and decidable and undecidable problems with examples. U April/May 2015
20. Explain the class P and NP problems with suitable example. U April/May 2015
21. Prove that “MPCP reduce to PCP”. A Nov/Dec 2015
22. Discuss about the tractable and intractable problems. U Nov/Dec 2015
23. State and explain RICE theorem. U Nov/Dec 2015
24. Describe about Recursive and Recursively Enumerable languages with examples. U Nov/Dec 2015
25. What is Universal Turing Machine? Bring out its significance. Also construct a TM to add two numbers and encode it. U May-June 2016
26. What is post correspondence problem (PCP) Explain with the help of an example. U May-June 2016, Nov-Dec2018
27. Elaborate on primitive recursive functions with an example. U Nov-Dec2016
28. Compare recursive language with recursively enumerable languages. AN Nov-Dec2016
29. What are tractable problems? Compare with intractable problems. AN Nov-Dec2016
30. Outline the concept of polynomial time reductions. U Nov-Dec2016
31. Explain recursive and recursively enumerable languages with suitable example. U May-June2107
32. Explain tractable and intractable problem with suitable example. U May-June2107
33. Explain universal TM. U Nov-Dec2017
34. Explain how to measure and classify complexity. U Nov-Dec2017
35. If L and its complement are recursively enumerable language, prove that L is recursive. E Nov-Dec 2018