GATE

Subject : CS 2008 - SOLUTIONS

(Q. NO. 1 – 20) 1 MARKS

1. $\lim_{x\to\infty} \dfrac{x-\sin x}{x+\cos x}$

$= \lim_{x\to\infty} \dfrac{x\left(1-\dfrac{\sin x}{x}\right)}{x\left(1+\dfrac{\cos x}{x}\right)}$

$= \lim_{x\to\infty} \dfrac{1-\dfrac{\sin x}{x}}{1+\dfrac{\cos x}{x}}$

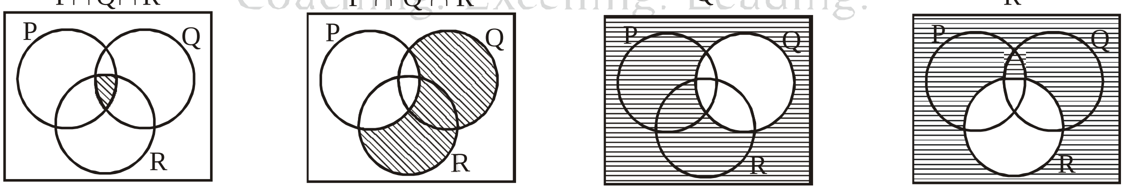
$= \dfrac{1-0}{1+0}$

$= 1$

(A) is the answer.

2. Using Venn diagrams

$P\cap Q\cap R$ $\quad$ $P^c\cap Q\cap R$ $\quad$ Q^c $\quad$ R^c

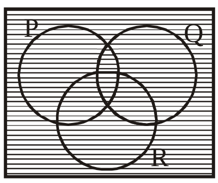


(I) (II) (III) (IV)

The union of these partial sets is the required result i.e.

(I) $\cup$ (II) $\cup$ (III) $\cup$ (IV)

$\therefore \quad (P \cap Q \cap R) \cup (P^C \cap Q \cap R) \cup Q^C \cup R^C = U$



(D) is the answer.

3. Since the given system has a unique solution, rank (A) = rank (A : B) = number of unknowns = 3

$$\therefore \quad (A : B) = \left[\begin{array}{ccc|c} 1 & 1 & 2 & 1 \\ 1 & 2 & 3 & 2 \\ 1 & 4 & \alpha & 4 \end{array}\right]$$

$R_2 \to R_2 - R_1$
$R_3 \to R_3 - R_1$

$$\therefore \quad (A : B) \sim \left[\begin{array}{ccc|c} 1 & 1 & 2 & 1 \\ 0 & 1 & 1 & 4 \\ 0 & 3 & \alpha - 2 & 3 \end{array}\right]$$

$R_3 \to R_3 - 3R_2$

$$\therefore \quad (A : B) \sim \left[\begin{array}{ccc|c} 1 & 1 & 2 & 1 \\ 0 & 1 & 1 & 1 \\ 0 & 0 & \alpha - 5 & 0 \end{array}\right]$$

Since rank of (A : B) must be 3,

$\alpha - 5 \neq 0$

$\therefore \quad \alpha \neq 5$

i.e. α can take any real value other than 5.

$\therefore$ None of the options is correct.

4. The given hexadecimal value is the representation of the special value +0.

exponent	Fraction	Number
All zero	All zero	Positive or negative zero

(D) is the answer.

5.

cd \ ab	00	01	11	10
00	1	1		1
01				
11				
10	1	1		X

$\therefore \quad f = \bar{a}\,\bar{d} + \bar{b}\,\bar{d}$

(A) is the answer.

6. $\sqrt{(121)_r} = (11)_r$

As per the rules, r > 2

$\therefore$ $\sqrt{r^2 + 2r + 1} = r + 1$

Squaring both sides,

$\therefore$ $r^2 + 2r + 1 = (r + 1)^2$

Given that both sides are equal, any value of r > 2 will do

(D) is the answer.

7. We can do a DFS or a BFS for finding the number of connected components in an undirected graph.

And the most efficient algorithm to perform a DFS or BFS takes time as $\theta(m + n)$.

where m – number of edges and

n – number of vertices.

(C) is the answer.

8. $f = f_1 f_2 + f_3$

$\therefore$ $\Sigma m(1, 6, 8, 15) = \Sigma m(4, 5, 6, 7, 8).\ f_2 + \Sigma m(1, 6, 15)$

$\therefore$ We need to find f_2 such that we get f as (1, 6, 8, 15)

(1, 6, 15) we already have from f_3

$\therefore$ f_2 should be chosen such that $f_1.f_2$ gives us 8 and any or all of (1, 6, 15)

$\therefore$ Among the given options, (6, 8) is a valid choice for the function f_2

(C) is the answer.

9. $L = \{a^p \mid p \text{ is a prime}\}$

L is not a regular language

$\therefore$ (B) is false.

L cannot be recognized by a PDA

$\therefore$ (C) is false.

L is a recursively enumerable language and can be accepted by a turing machine.

(D) is the answer.

10. (I) is true. Intersection of two regular languages is a regular language. And it is decidable if a regular language is infinite.

(II) is undecidable.

(III) is undecidable

(IV) is decidable.

(B) is the answer.

11. In complier design, a "handle" of a string is a substring that matches the RHS of a production and whose reduction to the non-terminal (on the LHS of the production) represents one step along the reverse of a rightmost derivation toward reducing to the start symbol.

Consider the following grammar G:

$S \to aAw$

$A \to B$

For the string aBw to reduce to the start symbol.

aBw

$\because$ $A \to B$ i.e. B is a handle of a aBw

= aAw

= S

Option (D) correctly describes a handle

(D) is the answer.

12. In compilers, code optimization is carried out on intermediate code other than machine code because intermediate code is machine independent unlike the machine code. So optimization can be carried out without considering into account the machine on which the code is going to execute.

Thus it supports portability of the code on other compilers. This is the main purpose to carry out code optimizations on intermediate code to enhance portability of the compiler.

(A) is the answer.

13. We know that recursively enumerable languages are not closed under complementation.

Therefore, L and $\overline{L}$ cannot both be guaranteed to be recursively enumerable unless L is recursive as recursive languages are closed under complementation.

(D) is the answer.

14. There is no restriction on the maximum size of data in the application layer. Hence, any size of data can be passed by the application layer on to the TCP layer.

(A) is the answer.

15. Following rules hold true for tuple relational calculus:

(i) $\forall t(P(t)) = (\neg\exists t)(\neg P(t))$

(ii) $\exists t(P(t)) = (\neg\forall t)(\neg P(t))$

$\therefore$ For the given statement,

$\forall t \in r(P(t)) \equiv \neg\exists t \in r(\neg P(t))$

$\therefore$ (C) is the answer.

16. A clustered index should satisfy the following properties:
   (i) Indexing should be done on non-key attributes
   (ii) The records should be ordered in the same manner the records are stored in the physical table.

   Hence, a clustered index must be non-key and ordering

   (A) is the answer.

17.
   - socket ( ) creates an endpoint for communication and returns a file descriptor for the socket
   - bind ( ) assigns a socket to an address.

     When a socket is created using socket ( ), it is only given a protocol family, but not assigned an address. This association with an address is performed with the bind ( ) system call before the socket can accept connections to other hoss.
   - After a socket has been associated with an address, listen ( ) prepares it for incoming connections.
   - When an application is listening for stream oriented connections from other hosts, it is notified of such events and must initialize the connection using the accept ( ) function. The accept ( ) function returns the new socket descriptor for the accepted connection, or –1 if an error occurs.
   - Using a connection oriented protocol like TCP, the connect ( ) system call establishes a connection A three-way handshake is performed which results in the sending of SYN packets.

     (D) is the answer.

18. The given line of code selects the maximum value out of x, y and z and assigns it to a.

   ∴ $x = 3$, $y = 4$ and $z = 2$ is the perfect combination that makes the variable a get the value 4.

   (A) is the answer.

19. BFS (Breadth first search) is a graph traversal technique that starts at a particular node (or any arbitrary node of a graph) and then explores the neighbour nodes first, before moving to the next level neighbours. BFS makes use of queue data structure.

   Suppose for the given graph, BFS starts at the node Q.

   ∴ It visits Q and the queue initially empty now contains Q.

Q			.....

   To explore the neighbours of Q, it will be dequeued from the queue and its neighbours will be enqueued which are M,N, P.

   ∴ Content of the queue = {M, N, P}

M	N	P		.....

   ∴ Visiting order as of now = QMNP

   Since, no more neighbours of Q are unvisited, we dequeue each element one by one to further explore its neighbours.

   Now we dequeue M (since it is at the front) and its unvisited neighbour R is added to the queue.

∴ Content of the queue is as follows:

~~M~~	N	P	R	.....

dequeued

∴ Visiting order as of now = QMNPR

Now, we dequeue N and search for its unvisited neighbour which is O. O is then added to the queue.

∴ Content of the queue now:

	~~N~~	P	R	O	

dequeued

∴ Visiting order = QMNPRO

As we can see there are no more unvisited neighbours hence all the elements of the queue are dequeued and the order is printed while the BFS stops.

∴ Order of visiting the graph using BFS is QMNPRO.

(C) is the answer.

20. The unix file system uses an extension of indexed allocation. It makes use of direct blocks, single indirect blocks, double indirect and triple indirect blocks.

(D) is the answer.

TECHNICAL SECTION (Q. NO. 21 – 75) 2 MARKS

21. Trapezoidal rule error is calculated as

$$E_n = \frac{-(b-a)^3}{12N^2} f''(c)$$

Maximum error = $\frac{1}{3}\times 10^{-6}$ (given)

∴ $$|E_n| < \frac{1}{3}\times 10^{-6}$$

∴ $$N^2 > \frac{(b-a)^3}{12\times\frac{1}{3}\times 10^{-6}}|f''(c)|$$

It is given that a = 1 and b = 2

∴ $$N > \frac{10^3}{2}\sqrt{|f''(c)|}$$

$$f'(x) = xe^x + e^x$$

$$f''(x) = xe^x + 2e^x$$

$f''(x)$ is maximum at $x = 2$

$$\therefore \quad f''(2) = f''(c) = 4e^2$$

$$\therefore \quad N > \frac{10^3}{2}\sqrt{4e^2}$$

$$\therefore \quad N > \frac{10^3}{2} \times 2e$$

$$\therefore \quad N > 1000e$$

(A) is the answer.

22. According to Newton-Raphson method,

$$x_{n+1} = x_n - \frac{f(x_n)}{f'(x_n)} \quad .....(i)$$

So the given equation is

$$x_{n+1} = \frac{1}{2}\left(x_n + \frac{R}{x_n}\right)$$

$$= \frac{x_n}{2} + \frac{R}{2x_n}$$

$$= x_n - \frac{x_n}{2} + \frac{R}{2x_n}$$

$$\therefore \quad x_{n+1} = x_n - \left(\frac{x_n^2 - R}{2x_n}\right) \quad .....(ii)$$

$\therefore$ On comparing equations (i) and (ii),

$$f(x_n) = x_n^2 - R$$

So root of $f(x_n)$ means

$$x_n^2 - R = 0$$

$$\therefore \quad x_n^2 = R$$

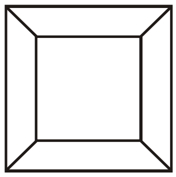
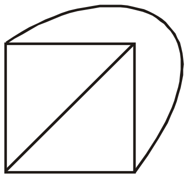
$$\therefore \quad x_n = \sqrt{R}$$

Hence, here we are computing square root of R.

(C) is the answer.

23. A graph is said to be planar if it can be drawn in a plane without any pair of edges crossing each other.

Examples of planar graph are

(A) is false as a disconnected graph can be planar i.e. drawn in a single plane.

(B) is false as Eulerian graph need not be planar. An undirected graph is Eulerian if all vertices have even degree.

(C) is true

(D) is false.

(C) is the answer.

24. P gives the sum of odd integers from 1 to 2k

Q gives the sum of even integers from 1 to 2k

Let $k = 4$

$\therefore \quad P = 1 + 3 + 5 + 7 = 16$

$Q = 2 + 4 + 6 + 8 = 20 = 16 + 4 = P + k$

$\therefore \quad Q = P + K$

$\therefore \quad P = Q - K$

(A) is the answer.

25. Let $f(x) = 3x^4 - 16x^3 + 24x^2 + 37$

Now, let's find the stationary points of f(x)

i.e. $f'(x) = 0$

$12x^3 - 48x^2 + 48x = 0$

$12x(x^2 - 4x + 4) = 0$

$12x(x - 2)^2 = 0$

$\therefore \quad x = 0, 2, 2$

Now, let us examine each point

$f''(x) = 36x^2 - 96x + 48$

$f''(x)$ at $x = 0$

$f''(0) = 48$ i.e. > 0

$\therefore$ There is a minima at $x = 0$ for f(x)

Now,

$f''(x)$ at $x = 2$

$f''(2) = 36(4) - 96(2) + 48$

$f''(2) = 0$

i.e. the second derivative test fails

$\therefore$ Check for concavity on either side of x = 2

$f''(1) = -12$

$f''(3) = 84$

i.e. the function is concave downwards to the left of x = 2 and concave upwards to its right.

f(x) changes its concavity at x = 2

$\therefore$ x = 2 is a point of inflection.

$\therefore$ f(x) has one extrema or more precisely a minima at x = 0

(B) is the answer.

26. $(P+\overline{Q})(P\overline{Q}+PR)(\overline{P}\overline{R}+\overline{Q})$

$= (P\overline{Q}+PR+P\overline{Q}+P\overline{Q}R)(\overline{P}\overline{R}+\overline{Q})$

$= (P(\overline{Q}+R+\overline{Q}+\overline{Q}R))(\overline{P}\overline{R}+\overline{Q})$

$= P(\overline{Q}(1+R)+R)(\overline{P}\overline{R}+\overline{Q})$

$= (P\overline{Q}+PR)(\overline{P}\overline{R}+\overline{Q})$

$= P\overline{Q}\overline{P}\overline{R}+P\overline{Q}+PR\overline{P}\overline{R}+PR\overline{Q}$

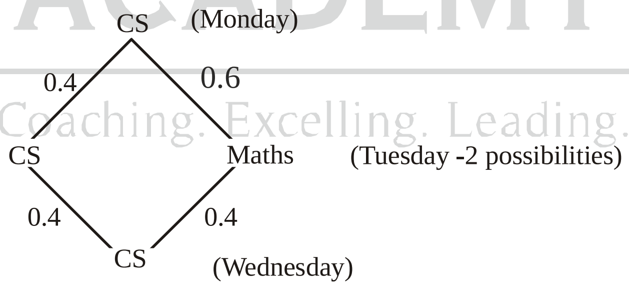
$= 0 + P\overline{Q}+0+PR\overline{Q}$

$= P\overline{Q}(1+R)$

$= P\overline{Q}$

(A) is the answer.

27. The probability tree will be as follows



We know that, if she studies computer science on a day, then the probability that she studies maths the next day is 0.6.

$\therefore$ Probability that she studies CS the next day

$= 1 - 0.6 = 0.4$

$\therefore$ Required probability = CS – CS – CS + CS – Maths – CS

$= 0.4 \times 0.4 + 0.6 \times 0.4$

$= 0.16 + 0.24$

$= 0.4$

(C) is the answer.

28. Consider the given matrices one by one

Let $A = \begin{bmatrix} 1 & 0 \\ 0 & 0 \end{bmatrix}$

$\therefore$ To find eigen values,

$$|A - \lambda I| = 0$$

$$\begin{vmatrix} 1-\lambda & 0 \\ 0 & -\lambda \end{vmatrix} = 0$$

$$-\lambda(1 - \lambda) = 0$$

i.e. $\lambda = 0$ or $\lambda = 1$

$\therefore$ It has eigen value as 1

Let $A = \begin{bmatrix} 0 & 1 \\ 0 & 0 \end{bmatrix}$

$\therefore$ $|A - \lambda I| = 0$

$$\begin{vmatrix} -\lambda & 1 \\ 0 & -\lambda \end{vmatrix} = 0$$

$$\lambda^2 = 0$$

$\therefore$ $\lambda = 0$

$\therefore$ It does not have eigen values as 1.

Let $A = \begin{bmatrix} 1 & -1 \\ 1 & 1 \end{bmatrix}$

$\therefore$ $|A - \lambda I| = 0$

$$\begin{vmatrix} (1-\lambda) & -1 \\ 1 & (1-\lambda) \end{vmatrix} = 0$$

$$1 - 2\lambda + \lambda^2 + 1 = 0$$

$$\lambda^2 - 2\lambda + 2 = 0$$

Here a (i.e. coefficient of λ^2) > 0

and $D = b^2 - 4ac = (-2)^2 - 4(2)(1) = 4 - 8 = -4$

$\therefore$ Roots are imaginary

So it does not have eigen value as 1.

Let $A = \begin{bmatrix} -1 & 0 \\ 1 & -1 \end{bmatrix}$

$\therefore$ $|A - \lambda I| = 0$

$$\begin{vmatrix} -1-\lambda & 0 \\ 1 & -1-\lambda \end{vmatrix} = 0$$

$$(-1-\lambda)^2 = 0$$

$$\lambda = -1, -1$$

$\therefore$ It does not have its eigen value as 1.

Hence, only one of the four matrices have the eigen value as 1.

(A) is the answer.

29. For the random variable X,

$$\mu = +1$$

$$\sigma_x^2 = 4$$

$$\therefore \quad \sigma_x = 2$$

Normalizing the variable X:

$$\therefore \quad Z = \frac{X-\mu_x}{\sigma_x} = \frac{-1-1}{2} = \frac{-2}{2} = -1$$

$\therefore$ $P(X \le -1) = P(Z \le -1) = P(Z \ge 1)$ For normal curve

For variable Y,

$$\mu_y = -1$$

$$\sigma_y^2 = ?$$

Normalizing the variable Y:

$$\therefore \quad Z = \frac{Y-\mu_y}{\sigma_y} = \frac{2-(-1)}{\sigma_y} = \frac{3}{\sigma_y}$$ on comparing we have,

$$P(Z \ge 1) = P\left(Z \ge \frac{3}{\sigma_y}\right)$$

$$\therefore \quad \frac{3}{\sigma_y} = 1$$

$$\sigma_y = 3$$

(A) is the answer.

30. The given statement states that for each FSA, there is an equivalent PDA.

In first order logic, it can be written as:

$$(\forall x\, fsa(x)) \Rightarrow (\exists y\, pda(y) \wedge equivalent(x, y))$$

(A) is the answer.

31. (I) $P \vee \sim Q$

(II) $\sim(\sim P \vee Q) = \sim(\sim P) \vee Q = P \vee \sim Q$

(III) $(P \wedge Q) \vee (P \vee \sim Q) \vee (\sim P \wedge \sim Q)$

$= P \wedge (Q \vee \sim Q) \vee (\sim P \wedge \sim Q)$

$= P \vee (\sim P \wedge \sim Q)$

$= P \vee \sim Q$

(IV) $(P \wedge Q) \vee (P \wedge \sim Q) \wedge (\sim P \wedge Q)$

$= P \wedge (Q \vee \sim Q) \vee (\sim P \wedge Q)$

$= P \wedge (\sim P \vee Q)$

$= P \vee Q$

$\therefore$ (I), (II) and (III) are equivalent.

(B) is the answer.

32. In a magnetic disk, seek times are not linear compared with the seek distance travelled because of the acceleration and deceleration of the arm, which is nothing but the arm starting and stopping inertia.

(B) is the answer.

33. (I) is false. Auto-increment addressing mode is not used in self relocating code.

(II) is false as there is no need of ALU for effective address calculation.

(III is true as in auto-increment addressing mode, the next data block is accessed depending on the size of a single data item.

$\therefore$ (C) is the answer.

34. RFE (Return From Exception) is a privileged trap instruction that is executed when an exception occurs, so that an exception is not allowed to execute.

(D) is the answer.

35. (I) is incorrect as content need not be exactly at the same point of time and hence we can also use write back policy.

(II) is false

(III) is false. Associativity of L2 can at the most be equal to L1

(IV) is necessarily true. L2 cache must be as large as L1

(A) is the answer.

36. (I) is false as by passing cannot handle all RAW hazards. It cannot handle LOAD instruction. The result of LOAD instruction is available after 4th cycle (MEM) and not EX (3rd cycle). Hence the pipeline will be stalled for one cycle.
    (II) is true. Register renaming can eliminate all WAR hazards.
    (III) is false. Branch prediction cannot completely eliminate all control hazards.
    (B) is the answer.

37. (I) is true as multiple register windows eliminate the need for memory accesses.
    Instead, we can use register accesses.
    (II) is false as memory accesses cannot be reduced here for register saves and restores for every variable.
    (III) is also false as there is no change in the number of memory accesses for instruction fetch using multiple register windows.
    (A) is the answer.

38. TLB is a small hardware which consists of a few entries of page table. Access time of TLB is comparatively less than the access time of page table. Whenever a page number is needed for the corresponding frame number, a TLB look up is done for that page first and then it is searched in the page table when there is a TLB miss.
    Hence TLB is accessed during effective address calculation.
    (B) is the answer.

39. Order of growth of functions is:
    $h(n) < f(n) < g(n)$
    $\therefore$ From the given options, we can easily conclude (D) is correct.
    $h(n) = O(f(n))$
    as $h(n) \le c.g(n)$ for some constant $c > 0$
    $g(n) = \Omega(f(n))$
    as $g(n) \ge c.f(n)$ for some constant $c > 0$
    (D) is the answer.

40. The best method to find if an integer appears more than n/2 times in a sorted array of n integers is to use binary search.
    Even in the worst case, time complexity will be O(log n)
    (B) is the answer.

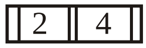
41. For maximum number of splits, we will have to consider the 10 successive insertions in such a way that the keys are in ascending order.

Let the keys be {2, 4, 6, 8, 10, 12, 14, 16, 18, 20}

order of B-tree = 4

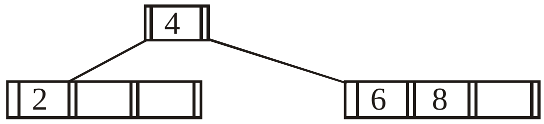
Insert 2

Insert 4

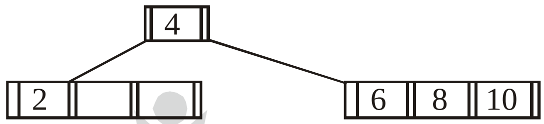


Insert 6

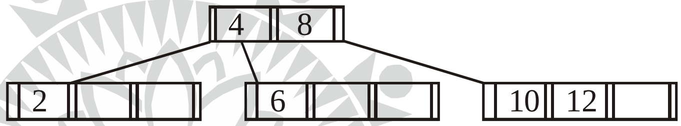
Insert 8 (1st split)



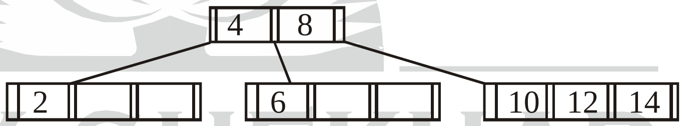
Insert 10



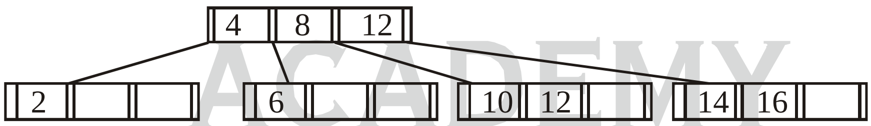
Insert 12 (2nd split)



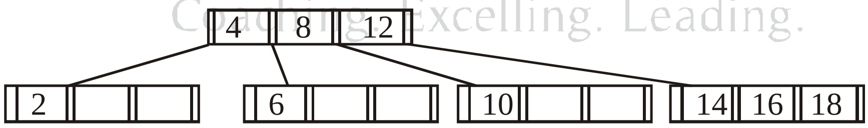
Insert 14



Insert 16 (3rd split)

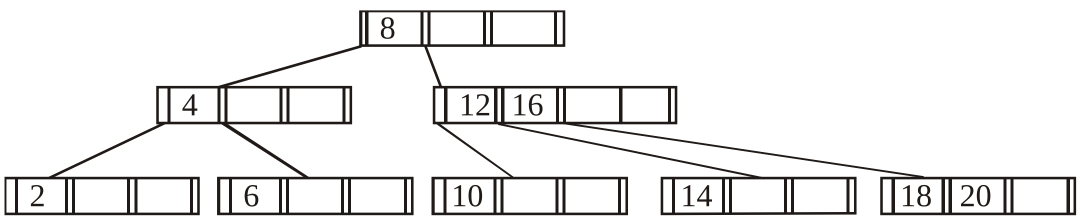


Insert 18



Insert 20 (2 splits: 1 leaf node split + 1 root node split)

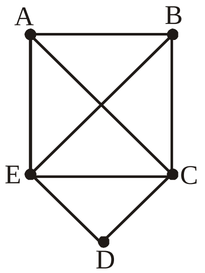
i.e. 4th and 5th split.



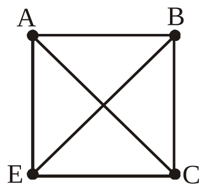
∴ Maximum number of node splitting operations = 5

(C) is the answer.

42. Consider the graph G



It has n = 5 vertices and 2n – 2 = 8 edges.

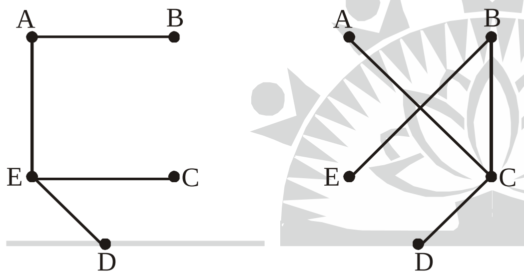


(A) is true. for the subgraph with k = 4 vertices,

we have 6 edges i.e. atmost (2k – 2)

(B) is true. Any minimum cut in G has atleast 2 edges.

(C) is true. The two edge disjoint spanning trees of G will be



We know that a spanning tree spans all the vertices in a graph. So, two edge disjoint spanning trees of a graph imply that every pair of vertices in that graph have two edge disjoint paths.

(D) is not always true.

∴ (D) is the answer.

43. One of the sublist contains at least one fifth of the elements. Hence to sort this list, we will require time T(n/5).

Now to sort the remaining sublist, we will require time T(4n/5).

Thus the total complexity of Quick sort can written as

$T(n) <= T(n/5) + T(4n/5) + n$

The ‘less than’ case will be considered when the smaller sublist contains more than (n/5) elements.

∴ The larger sublist will have less than (4n/5) elements. So in order to balance the recursion tree we consider ‘<’ case.

(B) is the answer.

44. Sum of subset problem is NP-complete

∴ (C) and (D) are true because a problem is said to be NP-complete if it belongs to both class NP and class NP-hard.

(A) is true. When the input is encoded in unary, length of the input is equal to value of input.

∴ Time complexity = O(nW) where W is the integer sum.

n and W are linear multiples of the length of the inputs. Hence, the time complexity is polynomial.

(B) is false. When the input is encoded in binary, length of W will be be logW

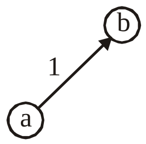
∴ for W = 4096, input length will be merely 12.

Hence O(nW) now becomes exponential in terms of the input lengths. So it is not a polynomial time algorithm.

(B) is the answer.

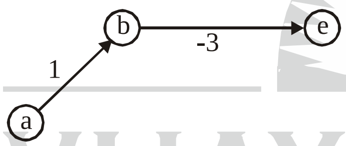
45. Dijkstra's single source shortest path algorithm is not guaranteed to work for negative edge weights. But it works for the given graph. Lets see how. Start from vertex a

a – b = 1 is the minimum.



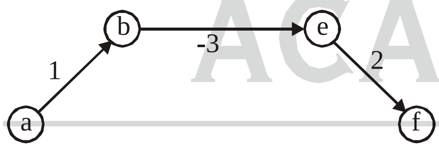
Now we have a to c = 3 and a to e = 1 – 3 = –2

∴ a to e is the minimum.



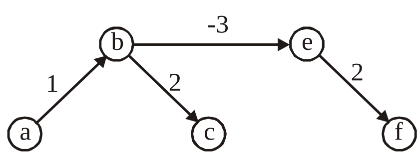
Now distances are a to c = 3 and a to f = 0

∴ a to f is the minimum.



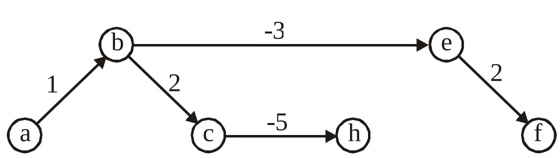
Now, we have to a to c = 3 and a to g = 3

Let's consider a to c



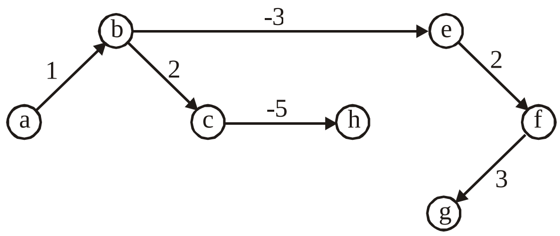
Now, we have a to g = 3 and a to h = –2

∴ a to h is the minimum.

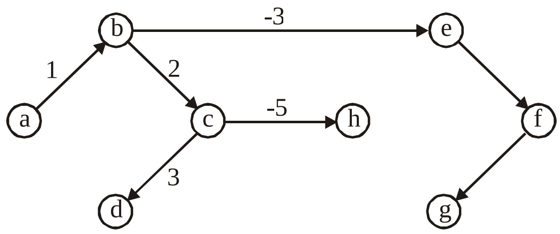


Now, we have at to g = 3 (a – b – e – f – g) and a to d = 6

∴ a to g is the minimum.



Now we are left with a to d (a – b – c – d) = 6



So, it works for all the vertices.

(D) is the answer.

46. Now that in a binary search tree (not a binary tree), the in order traversal is obtained by writing all its elements in an increasing order.

Thus, we now have post order traversal as well as in order traversal of a binary search tree. So we can determine a unique tree in O(n) time using the most efficient algorithm.

(B) is the answer.

47. Inserting 'n' more elements in an n-element heap is same like creating a heap with 2n elements.

We know that,

To create a heap with n-elements,

T.C. = O(n)

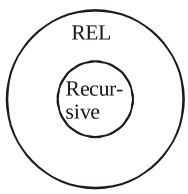
Similarly to create a heap with 2n-elements,

T.C. = $O(2n) = \theta(n)$

(B) is the answer.

48. (A), (B) and (C) are true.

(D) is false



For a REL (Recursively Enumerable Language), the Turing machine may reject the string that does not belong to the language or it may result in an indefinite looping of it.

However in case of recursive languages, there is no concept of infinite looping. It directly rejects the string which does not belong to the language.

Thus, every subset of a recursively enumerable set is not recursive.

(D) is the answer.

49. Z x Y results in the following automata

		a	b
11	$\rightarrow$ P	12	22
12	S	11	21
21	Q	22	12
22	R *	21	11

In option (A), in the row of S, it should be P and Q and not Q and P.

So, (A) is the answer.

50. (I) is true

(II) is false. We cannot remove ε from a CGF if the grammar itself contains ε

(III) is true. It is the definition of regular grammar.

(IV) is also true.

(C) is the answer.

51. H–S as only S gives even length patidromes

F – P as we know for a function, the number of formal parameters are same as the number of actual parameters.

There is only one option with the choices H–S and F–P i.e. (C)

Also, we know that as

$\therefore$ L = {wcw | w $\in$ (a|b)*} will check if w that has occurred earlier is present afterwards

(C) is the answer.

52. L(P) = { ε, 00, 001, 00110, 0001,.......}

Regular expression for P can be written as

0(00 + 01*1)* 01* + ε (as start state = final state)

$\therefore$ P – 1

L(Q) = { ε, 00, 0110, 01010, 010010, ...}

Regular expression for Q can be written as

0(10*1 + 00)*0 + ε (as start state = final state)

$\therefore$ Q – 2

L(R) = { ε, 01, 01011, 010011, 0101, ...}

Regular expression for R can be written as

0(10*1 + 10)*1 + ε (as start state = final state)

$\therefore$ R – 3

L(S) = { ε, 01, 010, 0100, 01011, 01010.....}

Regular expression for S can be written as

0(10*1 + 10)* 10* + ε (as start state = final state)

$\therefore$ S – 4

(C) is the answer.

53. (I) $\{a^n b^m \mid n \geq 0, m \geq 0\}$

It is a regular language. Regular expression is a*b*

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

It is a CFL as there is relation between the two alphabets a and b.

(III) $\{a^n b^m \mid n \neq m\}$

It is a CFL where equal number of a's followed by equal number of b's are not included in the language i.e. again a relation between a and b.

(IV) $\{xcy \mid x, y \in \{a, b\}^*\}$

It is a regular language. Regular expression is:

$(a + b)^* c (a + b)^*$

∴ (A) is the answer.

54. (I) is false as recursion cannot be implemented with static storage allocation.

(II) is true. If the programming language supports nesting of procedures, then there is a need of stack frames to arrange the activation records that it points to the latest call of the function.

(III) is false. Recursion can be implemented with dynamic storage allocation.

(IV) is false. Nesting of procedures are implemented using a stack-based allocation and not a heap-based.

(V) is true as in a stack based allocation, once a function is returned, it is removed from the function call stack. Hence, a function cannot seem to return a function using stack-based allocation.

(A) is the answer.

55. Both LR(1) and LALR(1) parsers use LR(1) sets of items to construct their passing tables.

Item sets of LALR(1) parsing table are formed from the item sets of LR(1) passing table.

These LR(1) item sets which have a same first component and differentiating in their look ahead symbols are merged together to form item sets of LALR(1).

Hence, LALR(1) parser for a grammar G can have S-R conflicts if and only if the LR(1) parser for G has S-R conflicts.

(B) is the answer.

56. In TCP congestion control algorithm, the size of the congestion window increases exponentially in the slow start phase upto the threshold.

After the threshold is reached, congestion avoidance phase is started and the congestion window increases linearly.

But in slow start phase, the congestion, window increases exponentially.

(D) is the answer.

57. Default mask of class B = 255.255.0.0

Subnet mask of class B = 255.255.248.0

Consider the third & fourth octect

$$248.0 \quad \underbrace{11111}_{\text{subnet bits}}\ \underbrace{000.00000000}_{\text{host bits}}$$

$\therefore$ Number of hosts per subnet = $2^{11} - 2 = 2046$

We do not consider the first address and last address of hosts because they are used as subnet address and broadcast address respectively.

(C) is the answer.

58. Initial capacity (C) of the bucket = 16Mb

(R) Input rate = 2 Mbps

(M) Output rate = 10 Mbps

Let 't' be the maximum duration for which the computer can transmit at full 10 Mbps.

Now,

$$\text{Input rate} = \text{Output rate}$$
$$C + Rt = Mt$$
$$\therefore \quad C = Mt - Rt$$
$$\therefore \quad C = (M - R)t$$
$$\therefore \quad t = \frac{C}{M - R}$$
$$= \frac{16\,\text{Mb}}{(10 - 2)\text{Mbps}}$$
$$= \frac{16}{8}\ \text{sec}$$
$$\therefore \quad t = 2\ \text{sec}$$

(B) is the answer.

59. Since accept ( ) system call is not executed by the server process, hence the connect ( ) system call will not get any response. So it won't wait forever but rather returns an error message to the server.

(C) is the answer.

60. c = 4; c = [~~4~~ ~~5~~ 7] 110

b = &c; b = [110] 120

a = &b; a = [120] 130

f(c, b, a)

Here, c is passed by value b is a pointer pointing to c and a is a pointer pointing to the pointer b

```
int f(int x, int *py, int **ppz)
{
    int y, z;
    *ppz + = 1;        // Here c = 4 + 1 = 5
    z = *ppz;          ∴ z = 5
*py + = 2;             // Here c = 5 + 2 = 7
y = *py;               ∴ y = 7
x + = 3;               // Here x = 4 + 3 = 7
return x + y + z; }    (As x is passed by value. So it will refer to its initially passed value)
```

So return value will be

$x + y + z = 7 + 7 + 5 = 19$

(B) is the answer.

61. To print the input string in reverse order, the given code makes use of recursion.

if the string is not empty i.e. ((c = getchar( )) ! = '\n')

then call reverse function

else put the characters i.e. putchar(c)

(D) is the correct choice.

62. The given function takes input as a linked list and swaps each two adjacent elements in the list.

For the given input 1, 2, 3, 4, 5, 6, 7

Output will be = 2, 1, 4, 3, 6, 5, 7

(B) is the answer.

63. For $X_b = 1$ and $Y_b = 0$, the implementation of semaphore operations P(S) and V(S) will work correctly.

Consider the signal operation V(S)

$P_b(X_b)$; $X_b = 0$

$S = S + 1$; // release a process

if ($S < = 0$) // i.e. if a process is in a blocked staten

$V_b(Y_b)$; then wake up $\therefore\ Y_b = 1$

$V_b(X_b)$; // to simply signal the process i.e. $X_b = 1$

Consider the wait operation P(S)

$P_b(X_b)$; $X_b = 0$

$S = S - 1$; // perform down operation i.e. block a process.

if ($S < 0$)

{

$V_b(X_b)$; // $S = -2$ i.e. there are already 2 processes in wait state, then we can let one of the process wake up by using these operations

$P_b(Y_b)$;

}

else

$V_b(X_b)$ // else if $S > = 0$, then wake up the currently blocked process.

(C) is the answer.

64. In synchronous I/O, the process performing I/O operation will be placed in blocked state till the I/O operation is executing. An ISR will be executed after the completion of I/O operation and the process will go from blocked state to ready state.

In asynchronous I/O, handler function will be registered while performing the I/O operation. The process will not be placed in the blocked state and the process continues to execute the remaining instructions. When the I/O operation is completed, a signal mechanism is used to notify the process the availability of data.

(B) is the answer.

65. (A) is false. Deadlock prevention handles deadlock by ensuring that one of the 4 necessary conditions don't occur i.e. Mutual exclusion, Hold and wait, No preemption and Circular wait. In deadlock prevention, the request for resources may not be granted even if the resulting state is a safe state. If the resulting state is a safe state, then the deadlock may or may not occur. However, if the resulting state is unsafe, then deadlock will definitely occur.

$\therefore$ (A) is the answer.

66. When fork( ) system call is called 'n' times,

Total #processes created = 2^n

Total #child processes = $2^n - 1$ (excluding the parent process)

$\therefore$ (B) is the answer.

67. Virtual address = 32 bits

Physical address = 36 bits

Page size = 4 KB

$\therefore$ No. of bits to represent page size = 12 bits

$\therefore$ No. of bits required to represent page frame number = 36 – 12 = 24 bits.

So, the third level page table must have 24 bits for addressing the page frames.

No. of bits required to represent the page number of page table = 32 – 12 = 20 bits.

So these 20 bits will be divided into three partitions as we have three-levels of page tables.

Now, we will find the number of bits required to represent each level page table.

For a 3rd-level page table,

Size of the page table = Number of entries in a single third level page table × size of the entry

$= 2^9 \times 4B$ ($\because$ bits 12 – 20 give 9 bits for indexing into third level page table)

$= 2^{11}$ B

Number of third level page tables possible

$$= \frac{\text{Physical memory size}}{\text{Size of third level page table}}$$

$$= \frac{2^{36}}{2^{11}}$$

$= 2^{25}$

Each of the third level page tables will be addressed into second level page table.

$\therefore$ Number of bits for addressing second-level page tables = 25 bits.

Similarly, we need to calculate the number of second level page tables and address each of them in the first level page table.

For a 2-nd level page table,

Size of the page table = Number of entries in a single second level page table × size of the entry.

$= 2^9 \times 4B$ ($\because$ bits 21 – 29 give 9 bits for indexing into second level page table)

$= 2^{11}$ B

Number of second level page table possible

$$= \frac{\text{Physical memory size}}{\text{size of a second level page table}}$$

$$= \frac{2^{36}}{2^{11}}$$

$= 2^{25}$

$\therefore$ Number of bits for addressing first level page tables = 25 bits.

$\therefore$ (D) is the answer.

Note: the number of page tables at level '*l*' is equal to the number of entries in the level '*l* – 1'.

68. I and IV are equivalent. IV is same as natural join expressed using other operators.

Also III gives the same result as I and IV because key is {P, Q} in both the relations. It is same as projection of common tuples based on the common attributes i.e. the key here which is nothing but natural join.

However II does not give the same result as the filtering of tuples is based only on the attribute P here. So it will give different result.

(D) is the answer.

69. Consider the relation Book

{Author, Title} is the key

From the given functional dependencies, there is no such dependency as

Proper subset of key $\rightarrow$ Non-key attribute

$\therefore$ Book is in 2NF,

But we have

Non-key attribute $\rightarrow$ Non-key attribute

as Catalog-no $\rightarrow$ Publisher Year

$\therefore$ Book is not in 3NF. The highest normal form that if satisfies is of 2NF only.

Now, consider the relation Collection

{Author, Title} is the key

Again, there is no such dependency here as

Proper subset of key $\rightarrow$ non-key attribute

$\therefore$ Collection is in 2NF

Also there is no such dependency as

Non-key attribute $\rightarrow$ Non-key attribute

as collection has only one Non-key attribute i.e. catalog-no.

$\therefore$ Collection is in 3NF.

Also there is no such dependency as

Proper subset of one candidate key $\rightarrow$ Proper subset of another candidate key

as there is only one candidate key

$\therefore$ Collection is in BCNF.

From the given options, (C) is true.

70. Index will be built on < key, block pointer >

$\therefore$ Size of the index = 6 + 10 = 16B.

In the first level, index will be there for each record of the file.

$\therefore$ Size of first – level index = 16384 (records) × 16B

Number of blocks in first-level index

$$= \frac{\text{Size of first} - \text{level index}}{\text{Block size}}$$

$= \frac{16384 \times 16B}{1024\,B}$

$= 256$

In the second level, there will be an index entry for each block in the first level.

$\therefore$ Total entries in the first-level = number of blocks in the first level = 256.

$\therefore$ Size of second-level index = 256 × 16B

Number of blocks in second-level index

$= \frac{\text{Size of second} - \text{level index}}{\text{Block size}}$

$= \frac{256 \times 16B}{1024B}$

$= 4$

$\therefore$ (C) is the answer.

71. For set associative memory, format is

← 32 →

tag bits	set offset	Block offset

Block size = 16B

$\therefore$ Block offset $= \log_2 \lceil \text{Block size} \rceil$

$= \log_2 \lceil 16 \rceil$

$= 4$ bits

Blocks in cache $= \frac{\text{Cache size}}{\text{Block size}}$

$= \frac{64\,KB}{16\,B}$

$= 4K$

$= 2^{12}$

#sets $= \frac{\#\text{blocks in cache}}{P - \text{way}}$

$= \frac{2^{12}}{2}$

$= 2^{11}$

$\therefore$ set offset $= \log_2 \lceil \#\text{sets} \rceil$

$= \log_2 \lceil 2^{11} \rceil$

$= 11$ bits

∴ Number of tag bits = 32 – (4 + 11)

$= 17$ bits

Size of cache tag directory = #tag bits *#blocks in cache

$= 17 \times 2^{12}$ bits

$= 69632$ bits

$= \frac{69632}{1024}$ K bits

$= 68$ K bits

(D) is the answer.

72. Size of each element in the array = 8B

Size of the array = #elements *size of each element

$= 1024 \times 1024 \times 8B$

$= 2^{23}$ B

Size of each block= 16B

∴ Number of blocks required for the array

$= \frac{2^{23}B}{16B}$

$= 2^{19}$ blocks

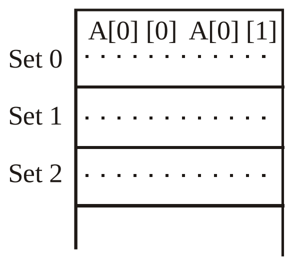
Number of elements in each block

$= \frac{16}{8}$

$= 2$ elements

As calculated in the previous question, number of blocks in cache = 2^{12}

and number of sets = $\frac{2^{12}}{2} = 2^{11}$



∴ Main memory block no. 0 will map to the set

$0 \bmod 2^{11} = 0$

Main memory block no.1 will map to the set

$1 \bmod 2^{11} = 1$

.

.

.

Main memory block no. 2048 will map to the set

$2048 \bmod 2^{11} = 0$

i.e. Main memory block no. 0 and 2048 map to the same set.

We know that each row of the array has 1024 elements.

$\therefore$ No. of blocks required = $\frac{1024}{2}$ = 512 blocks

Row 0 will have 0 – 511 blocks

Row 1 will have 512 – 1023 blocks

.

.

.

Row x will have 2048 – 2559 blocks

$\therefore$ $x = \frac{2048}{512} = 4$

$\therefore$ ARR [4] [0] has the same cache index as ARR[0] [0]

(B) is the answer.

73. Cache hit ratio = $\frac{\text{Number of hits}}{\text{Total accesses}}$

Here, every alternative memory access is a cache miss. As the cache block size is 16B and each block can accommodate only 2, elements ($\because$ size of each element = 8B), hence during a memory access, only next element gets filled in the cache.

$\therefore$ Miss rate = 50%

Which gives the hit rate as 50%

(C) is the answer.

74. For f1(n), a recursion tree is formed.

Time complexity for f1(n) will be given as

$T(n) = 2T(n - 1) + 3T(n - 2)$

The recurrence relation is same as for computing the fibonacci series which gives the solution as

$T(n) = O(2^n)$

For f2(n), all the recursive calls will be avoided. It saves the results in an array making use of dynamic programming.

$\therefore$ Time complexity of f2(n) = $\theta(n)$

(B) is the answer.

75. Both f1 and f2 are calculating the same function. f2 uses dynamic programming whereas f1 doesn't.

$$f1(0) = 0;\ f1(1) = 1$$
$$f1(2) = 2f1(1) + 3f1(0) = 2$$
$$f1(3) = 2f1(2) + 3f1(1) = 7$$
$$f1(4) = 2f1(3) + 3f1(2) = 20$$
$$f1(5) = 2f1(4) + 3f1(3) = 61$$
$$f1(6) = 2f1(5) + 3f1(4) = 182$$
$$f1(7) = 2f1(6) + 3f1(5) = 547$$
$$f1(8) = 2f1(7) + 3f1(6) = 1640 = f2(8)$$

(C) is the answer.

LINKED ANSWER QUESTION (Q. NO. 76 – 85) 2 MARKS

76. An instruction following a branch instruction in a pipeline is always executed irrespective of whether branch is taken or not. This is because usually branch target address is available at the ID phase which in turn introduces stall in the pipeline. This slot is called as branch delay slot. As we do not know, the outcome of the branch beforehand, we need to fill the delay slot with an appropriate instruction such as a NOP (no operation) instruction.

(A) is the answer.

77. I4 can occupy the delay slot. We can move the STORE instruction below the BRANCH instruction. Since there will be no dependency, the STORE instruction will be executed as the next instruction irrespective of whether branch is taken or not.

I2 and I3 cannot occupy delay slot as they directly depend on the branching instruction I4 whereas I1 indirectly depends on I4.

(D) is the answer.

78.

n (length of the string)	x_n	#strings
1	0, 1	2
2	01, 10, 11	3
3	010, 011, 101, 110, 111	5
4	0101, 0110, 0111, 1010, 1011, 1101, 1110, 1111	8

This shows that

$$x_3 = x_2 + x_1$$
$$x_4 = x_3 + x_2$$

Similarly, $x_n = x_{n-1} + x_{n-2}$

(D) is the answer.

79. $x_5 = x_4 + x_3$
$= 8 + 5$
$= 13$

There is no such option as 13. So, we select the closest option which is 16.
(D) is the answer.

80. For calculating X[i][j], we have two options:
(i) Include ai in the subset as $X[i - 1][j - a_i]$ OR
(ii) Do not include the value ai in the subset as X[i – 1][j]
$\therefore$ $X[i, j] = X[i - 1, j] \vee X[i, j - a_i]$
(B) is the answer.

81. If we get the entry X[n, W] as true then there is a subset of $\{a_1, a_2, ... a_n\}$ that has the sum as W.
(C) is the answer.

82. M and P are strong entity sets. Hence they must be represented by separate tables. But for relationship set R1, participation of M is total. So for many to one relationship that is total on the many side can be represented by adding an extra attribute on the ‘many’ side containing the primary key of the “one” side. In this way an additional table for R1 can be avoided.
Also N is a weak entity set and hence we need to include the primary key of the strong entity set P i.e. P1
This would result in the formation of following tables.
P (<u>P1</u>, P2)
M (<u>M1</u>, M2, M3, P1)
N (<u>P1, N1</u>, N2)
$\therefore$ Minimum of 3 tables are required.
(B) is the answer.

83. As explained in the previous question, option (A) gives the correct attribute set for the table M.
(A) is the answer.

84. The given program does not work when the element to be searched is the maximum element in the array Y(i.e. the last element) or greater than the maximum element. For such conditions, the program goes into an infinite loop as the while condition is always true.
(C) is the answer.

85. For correct implementation of the program, we should change the line 6 to

```
if (Y[k] < x)
    i = k + 1;    // search in the right half of the array
else
    j = k – 1;    // search in the left half of the array.
```

It is an implementation of binary search.
(A) is the answer.