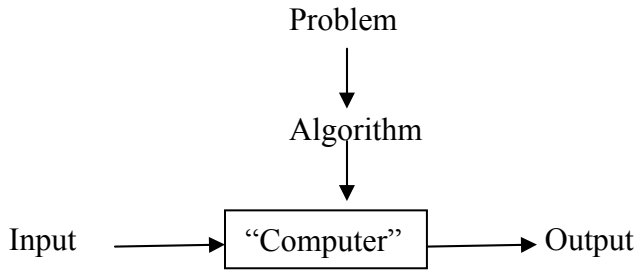
DESIGN AND ANALYSIS OF ALGORITHMS

UNIT I INTRODUCTION

Definition and properties of an algorithm- Analysis of algorithms. Divide and Conquer -The general method- Binary search- Finding maximum and minimum element- Analysis of Merge sort- Analysis of Quick sort- Analysis of Selection sort- Analysis of Heap sort

1. **What is an algorithm?**

An algorithm is a sequence of unambiguous instructions for solving a problem. i.e., for obtaining a required output for any legitimate input in a finite amount of time



2. **What do you mean by Amortized Analysis?**

Amortized analysis finds the average running time per operation over a worst case sequence of operations. Amortized analysis differs from average-case performance in that probability is not involved; amortized analysis guarantees the time per operation over worst-case performance.

3. **What are important problem types? (or) Enumerate some important types of problems.**

1. Sorting
2. Searching
3. Numerical problems
4. Geometric problems
5. Combinatorial Problems
6. Graph Problems
7. String processing Problems

4**. Name some basic Efficiency classes**

1. Constant
2. Logarithmic
3. Linear
4. nlogn
5. Quadratic
6. Cubic
7. Exponential
8. Factorial

5. **What are algorithm design techniques?**

Algorithm design techniques (or strategies or paradigms) are general approaches to solving problems algorithmatically, applicable to a variety of problems from different areas of computing. General design techniques are:

(i) Brute force	(ii) divide and conquer
(iii) decrease and conquer	(iv) transform and concquer
(v) greedy technique	(vi) dynamic programming
(vii) backtracking	(viii) branch and bound

6. **How is an algorithm's time efficiency measured**?

Time efficiency indicates how fast the algorithm runs. An algorithm's time efficiency is measured as a function of its input size by counting the number of times its basic operation (running time) is executed. Basic operation is the most time consuming operation in the algorithm's innermost loop.

7. **What is Big 'Oh' notation?**

A function t(n) is said to be in O(g(n)), denoted $t(n) \in O(g(n))$, if t(n) is bounded above by some constant multiple of g(n) for all large n, i.e., if there exist some positive constant c and some nonnegative integers n_0 such that

$$t(n) \leq cg(n) \text{ for all } n \geq n_0$$

8. **What is an Activation frame?**

It is a storage area for an invocation of recursive program (parameters, local variables, return address/value etc.). Activation frame allocated from frame stack pointed by frame pointer.

9. **Define order of an algorithm**

Measuring the performance of an algorithm in relation with the input size n is known as order of growth.

10. **What is recursive call?**

Recursive algorithm makes more than a single call to itself is known as recursive call. An algorithm that calls itself is direct recursive.An algorithm"A" is said to be indirect recursive if it calls another algorithm which in turn calls "A".

11. **What do you mean by stepwise refinement?**

In top down design methodology the problem is solved in sequence (step by step) is known as stepwise refinement.

12**. How is the efficiency of the algorithm defined?**

The efficiency of an algorithm is defined with the components.

(i) Time efficiency -indicates how fast the algorithm runs

(ii) Space efficiency -indicates how much extra memory the algorithm needs

13. **Define direct recursive and indirect recursive algorithms**.

Recursion occurs where the definition of an entity refers to the entity itself. Recursion can be direct when an entity refers to itself directly or indirect when it refers to other entities which refers to it. A (Directly) recursive routine calls itself. Mutually recursive routines are an example of indirect recursion. A (Directly) recursive data type contains pointers to instances of the data type.

14. **What are the characteristics of an algorithm?**

Every algorithm should have the following five characteristics

(i) Input

(ii) Output

(iii) Definiteness

(iv) Effectiveness

(v) Termination

Therefore, an algorithm can be defined as a sequence of definite and effective instructions, which terminates with the production of correct output from the given input. In other words, viewed little more formally, an algorithm is a step by step formalization of a mapping function to map input set onto an output set.

15. **What do you mean by time complexity and space complexity of an algorithm?**

Time complexity indicates how fast the algorithm runs. Space complexity deals with extra memory it require. Time efficiency is analyzed by determining the number of repetitions of the basic operation as a function of input size. Basic operation: the operation that contributes most towards the running time of the algorithm The running time of an algorithm is the function defined by the number of steps (or amount of memory) required to solve input instances of size n.

16. **Define Big Omega Notations**

A function t(n) is said to be in Ω(g(n)) , denoted t(n) Є Ω((g(n)) , if t(n) is bounded below by some positive constant multiple of g(n) for all large n, i.e., if there exist some positive constant c and some nonnegative integer n_0 such that

$$t(n) \geq cg(n) \text{ for all for all } n \geq n_0$$

17. **What are the different criteria used to improve the effectiveness of algorithm?**

(i) The effectiveness of algorithm is improved, when the design, satisfies the following constraints to be minimum.

Time efficiency - how fast an algorithm in question runs.

Space efficiency – an extra space the algorithm requires

(ii) The algorithm has to provide result for all valid inputs.

18. **Analyze the time complexity of the following segment:**

```
for(i=0;i<N;i++)
for(j=N/2;j>0;j--)
sum++;
```

Time Complexity= $N * N/2$

$= N^2 /2$

$Є O(N^2)$

19. **write general plan for analyzing non-recursive algorithms.**

i. Decide on parameter indicating an input's size.
ii. Identify the algorithm's basic operation
iii. Checking the no.of times basic operation executed depends on size of input.if it depends on some additional property,then best,worst,avg.cases need to be investigated
iv. Set up sum expressing the no.of times the basic operation is executed. (establishing order of growth)

20**. How will you measure input size of algorithms?**

The time taken by an algorithm grows with the size of the input. So the running time of the program depends on the size of its input. The input size is measured as the number of items in the input that is a parameter n is indicating the algorithm's input size.

21. **Define the terms: pseudocode, flow chart**

A pseudocode is a mixture of a natural language and programming language like constructs. A pseudocode is usually more precise than natural language. A flowchart is a method of expressing an algorithm by a collection of connected geometric shapes containing descriptions of the algorithm's steps.

22. **write general plan for analyzing recursive algorithms.**

- **i.** Decide on parameter indicating an input's size.
- **ii.** Identify the algorithm's basic operation
- **iii.** Checking the no.of times basic operation executed depends on size of input.if it depends on some additional property,then best,worst,avg.cases need to be investigated
- **iv.** Set up the recurrence relation,with an appropriate initial condition,for the number of times the basic operation is executed
- **v.** Solve recurrence (establishing order of growth)

23. **What do you mean by Combinatorial Problem?**

Combinatorial Problems are problems that ask to find a combinatorial object-such as permutation, a combination, or a subset--that satisfies certain constraints and has some desired property.

24. **Define Big Theta Notations**

A function t(n) is said to be in $\Theta(g(n))$, denoted $t(n) \in \Theta(g(n))$, if t(n) is bounded both above and below by some positive constant multiple of g(n) for all large n, i.e., if there exist some positive constants c1 and c2 and some nonnegative integer n_0 such that

$$c1\ g(n) \leq t(n) \leq c2g(n) \text{ for all } n \geq n0$$

25. What is performance measurement?

Performance measurement is concerned with obtaining the space and the time requirements of a particular algorithm.

26. What is an algorithm?

An algorithm is a finite set of instructions that, if followed, accomplishes a particular task. In addition, all algorithms must satisfy the following criteria:

1) input
2) Output
3) Definiteness
4) Finiteness
5) Effectiveness.

27. Define Program.

A program is the expression of an algorithm in a programming language. Sometimes works such as procedure, function and subroutine are used synonymously program.

28. What is recursive algorithm?

An algorithm is said to be recursive if the same algorithm is invoked in the body.

29. **What is space complexity and time complexity ?**

The space complexity of an algorithm is the amount of memory it needs to run to completion. The time complexity of an algorithm is the amount of computer time it needs to run to completion.

30. **Give the two major phases of performance evaluation**

Performance evaluation can be loosely divided into two major phases:
(i) a prior estimates (performance analysis)
(ii) a Posterior testing(performance measurement)

31. **Define input size.**

The input size of any instance of a problem is defined to be the number of words(or the number of elements) needed to describe that instance.

32. **Define best-case step count**.

The best-case step count is the minimum number of steps that can be executed for the given parameters.

33. **Define worst-case step count.**

The worst-case step count is the maximum number of steps that can be executed for the given parameters.

34. **Define average step count.**

The average step count is the average number of steps executed an instances with the given parameters.

35. **Define Little "oh".**

The function f(n) = 0(g(n)) iff

$$\lim_{n \to \infty} \frac{f(n)}{g(n)} = 0$$

36. **Define Little Omega.**

The function f(n) = ω (g(n)) iff

$$\lim_{n \to \infty} \frac{f(n)}{g(n)} = 0$$

37. **Write algorithm using iterative function to fine sum of n numbers**.

```
Algorithm sum(a,n)
{ S : = 0.0
For i=1 to n do
S : - S + a[i];
Return S;
}
```

38. **Write an algorithm using Recursive function to fine sum of n numbers,**

```
Algorithm Rsum (a,n)
{
```

If(n≤0) then
Return 0.0;
Else Return Rsum(a, n- 1) + a(n);

39. **Define the divide an conquer method.**

Given a function to compute on 'n' inputs the divide-and-comquer strategy suggests splitting the inputs in to'k' distinct susbsets, 1<k <n, yielding 'k' subproblems. The subproblems must be solved, and then a method must be found to combine subsolutions into a solution of the whole. If the subproblems are still relatively large, then the divide-and conquer strategy can possibly be reapplied.

40. **Define control abstraction**.

A control abstraction we mean a procedure whose flow of control is clear but whose primary operations are by other procedures whose precise meanings are left undefined.

Write the Control abstraction for Divide-and conquer.

```
Algorithm DAndC(P)
{
if small(p) then return S(P);
else
{
divide P into smaller instance p1,p2,…pk, k ≥1;
Apply DAndC to each of these subproblems
Return combine (DAnd C(p1) DAnd C(p2),----, DAnd (pk));
}
}
```

41.**What is the substitution method?**

One of the methods for solving any such recurrence relation is called the substitution method.

42. **What is the binary search?**

If 'q' is always chosen such that 'aq' is the middle element(that is, q=[(n+1)/2), then the resulting search algorithm is known as binary search.

43. **Give computing time for Binary search?**

In conclusion we are now able completely describe the computing time of binary search by giving formulas that describe the best, average and worst cases.

Successful searches

Θ(1)	Θ (logn)	Θ (Logn)
best	average	worst

Unsuccessful searches

Θ (logn)

best, average, worst

45. **Define external path length?**

The external path length E, is defines analogously as sum of the distance of all external nodes from the root.

46. **Define internal path length.**

The internal path length 'I' is the sum of the distances of all internal nodes from the root.

47. **What is the maximum and minimum problem?**

The problem is to find the maximum and minimum items in a set of 'n' elements. Though this problem may look so simple as to be contrived, it allows us to demonstrate divideand-conquer in simple setting.

48. **What is the Quick sort?**

Quicksort is divide and conquer strategy that works by partitioning it's input elements according to their value relative to some preselected element(pivot). it uses recursion and the method is also called partition –exchange sort.

49. **Write the Analysis for the Quick sort.**

O(nlogn) in average and best cases

$O(n^2)$ in worst case

50. **what is Merge sort? and Is insertion sort better than the merge sort?**

Merge sort is divide and conquer strategy that works by dividing an input array in to two halves,sorting them recursively and then merging the two sorted halves to get the original array sorted

Insertion sort works exceedingly fast on arrays of less then 16 elements, though for large 'n' its computing time is $O(n^2)$.

51. **Write a algorithm for straightforward maximum and minimum**?

```
Algorithm straight MaxMin(a,n,max,min)
//set max to the maximum and min to the minimum of a[1:n]
{
max := min: = a[i];
for i = 2 to n do
{
if(a[i] >max) then max: = a[i];
if(a[i] >min) then min: = a[i];
}
}
```

52. **what is general divide and conquer recurrence?**

Time efficiency T(n)of many divide and conquer algorithms satisfies the equation T(n)=a.T(n/b)+f(n).This is the general recurrence relation.

53. What is Master's theorem?

Theorem (Master Theorem)

Let $T(n)$ be a monotonically increasing function that satisfies

$$\begin{aligned} T(n) &= aT(\tfrac{n}{b}) + f(n) \\ T(1) &= c \end{aligned}$$

where $a \geq 1, b \geq 2, c > 0$. If $f(n) \in \Theta(n^d)$ where $d \geq 0$, then

$$T(n) = \begin{cases} \Theta(n^d) & \text{if } a < b^d \\ \Theta(n^d \log n) & \text{if } a = b^d \\ \Theta(n^{\log_b a}) & \text{if } a > b^d \end{cases}$$

54. **Write the algorithm for Iterative binary search?**

```
Algorithm BinSearch(a,n,x)
//Given an array a[1:n] of elements in nondecreasing
// order, n>0, determine whether x is present
{
low : = 1;
high : = n;
while (low < high) do
{
mid : = [(low+high)/2];
if(x < a[mid]) then high:= mid-1;
else if (x >a[mid]) then low:=mid + 1;
else return mid;
}
return 0;
}
```

55. **Describe the recurrence relation for merge sort?**

If the time for the merging operation is proportional to n, then the computing time of merge sort is described by the recurrence relation

$$T(n) = \begin{cases} a & n = 1, \quad a \text{ a constant} \\ 2T(n/2) + n & n > 1, \quad c \text{ a constant} \end{cases}$$

UNIT II GREEDY METHOD

The general method- Optimal storage on tapes- Knapsack problem- Minimum spanning trees- Single source shortest path method

1. **Explain the greedy method.**

   Greedy method is the most important design technique, which makes a choice that looks best at that moment. A given 'n' inputs are required us to obtain a subset that satisfies some constraints that is the feasible solution. A greedy method suggests that one can device an algorithm that works in stages considering one input at a time.

2. **Define feasible and optimal solution.**

   Given n inputs and we are required to form a subset such that it satisfies some given constraints then such a subset is called feasible solution. A feasible solution either maximizes or minimizes the given objective function is called as optimal solution

3. **Write the control abstraction for greedy method.**

```
Algorithm Greedy (a, n)
{
solution=0;
for i=1 to n do
{
x= select(a);
if feasible(solution ,x) then
solution=Union(solution ,x);
}
return solution;
}
```

4. **What are the constraints of knapsack problem?**

   To maximize $\sum_{1\le i\le n} p_i x_i$

   The constraint is : $\sum_{1\le i\le n} w_i x_i \ge m$ and $0 \le x_i \le 1 \quad 1 \le i \le n$

   where m is the bag capacity, n is the number of objects and for each object i w_i and p_i are the weight and profit of object respectively.

5. **What is a minimum cost spanning tree?**

A spanning tree of a connected graph is its connected acyclic subgraph that contains all vertices of a graph. A minimum spanning tree of a weighted connected graph is its spanning tree of the smallest weight where bweight of the tree is the sum of weights on all its edges.

A minimum spanning subtree of a weighted graph (G,w) is a spanning subtree of G of minimum weight $w(T) = \sum_{e \in T} w(e)$

Minimum Spanning Subtree Problem: Given a weighted connected undirected graph (G,w), find a minimum spanning subtree

6. **Specify the algorithms used for constructing Minimum cost spanning tree.**
   a) Prim's Algorithm
   b) Kruskal's Algorithm
7. **State single source shortest path algorithm (Dijkstra's algorithm).**

For a given vertex called the source in a weigted connected graph,find shotrtest paths to all its other vertices.Dijikstra's algorithm applies to graph with non-negative weights only.

8. **What is Knapsack problem?**

A bag or sack is given capacity and n objects are given. Each object has weight wi and
profit pi .Fraction of object is considered as xi (i.e) 0<=xi<=1 .If fraction is 1 then entire object is put into sack. When we place this fraction into the sack we get wixi and pixi.

9. **Write any two characteristics of Greedy Algorithm?**

* To solve a problem in an optimal way construct the solution from given set of candidates.
* As the algorithm proceeds, two other sets get accumulated among this one set contains
the candidates that have been already considered and chosen while the other set contains
the candidates that have been considered but rejected.

10. **What is the Greedy approach?**

The method suggests constructing solution through sequence of steps,each expanding partially constructed solution obtained so far,until a complete solution is reached. On each step,the choice must be

- Feasible(satisfy problem constraints)
- Locally optimal(best local choice among all feasible choices available on that step)
- Irrevocable(once made,it cant be changed)

11. What are the steps required to develop a greedy algorithm?

* Determine the optimal substructure of the problem.
* Develop a recursive solution.
* Prove that at any stage of recursion one of the optimal choices is greedy choice. Thus it is always safe to make greedy choice.
* Show that all but one of the sub problems induced by having made the greedy choice are empty.
* Develop a recursive algorithm and convert into iterative algorithm.

13. Define forest.

Collection of sub trees that are obtained when root node is eliminated is known as forest

14. Write the difference between the Greedy method and Dynamic programming.

Greedy method	Dynamic programming
Only one sequence of decision is generated.	Many number of decisions are generated.
It does not guarantee to give an optimal solution always.	It definitely gives an optimal solution always.

15.**state the requirement in optimal storage problem in tapes.**

Finding a permutation for the n programs so that when they are stored on the tape in this order the MRT is minimized.This problem fits the ordering paradigm.

16. **state efficiency of prim's algorithm.**

$O(|v|^2)$ (WEIGHT MATRIX AND PRIORITY QUEUE AS UNORDERED ARRAY)
O(|E| LOG|V|) (ADJACENCY LIST AND PRIORITY QUEUE AS MIN-HEAP)

17. State Kruskal Algorithm.

The algorithm looks at a MST for a weighted connected graph as an acyclic subgraph with |v|-1 edges for which the sum of edge weights is the smallest.

18. **state efficiency of Dijkstra's algorithm.**

$O(|v|^2)$ (WEIGHT MATRIX AND PRIORITY QUEUE AS UNORDERED ARRAY)
O(|E| LOG|V|) (ADJACENCY LIST AND PRIORITY QUEUE AS MIN-HEAP)

19. Differentiate subset paradigm and ordering paradigm

subset paradigm	ordering paradigm
At each stage a decision is made regarding whether a particular input is in an optimal solution (generating sub optimal solutions)	For problems that do not call for selection of optimal subset,in the greedy manner we make decisions by considering inputs in some order
Example kNAPSACK,MST	Optimal storage on tapes

UNIT III

DYNAMIC PROGRAMMING

The General method- All pairs shortest path- Optimal binary tree- Multistage graphs

UNIT-III

1. Write the difference between the Greedy method and Dynamic programming.

Greedy method	Dynamic programming
1.Only one sequence of decision is generated.	1.Many number of decisions are generated.
2.It does not guarantee to give an optimal solution always.	2.It definitely gives an optimal solution always.

2. Define dynamic programming.

Dynamic programming is an algorithm design method that can be used when a solution to the problem is viewed as the result of sequence of decisions. It is technique for solving problems with overlapping subproblems.

3. What are the features of dynamic programming?

- Optimal solutions to sub problems are retained so as to avoid recomputing of their values.
- Decision sequences containing subsequences that are sub optimal are not considered.
- It definitely gives the optimal solution always.

4. What are the drawbacks of dynamic programming?

- Time and space requirements are high, since storage is needed for all level.
- Optimality should be checked at all levels.

5. Write the general procedure of dynamic programming.

The development of dynamic programming algorithm can be broken into a sequence of 4 steps.

1. Characterize the structure of an optimal solution.
2. Recursively define the value of the optimal solution.
3. Compute the value of an optimal solution in the bottom-up fashion.
4. Construct an optimal solution from the computed information.

6. Define principle of optimality.

It states that an optimal solution to any of its instances must be made up of optimal solutions to its subinstances.

7. Define multistage graph

A multistage graph G =(V,E) is a directed graph in which the vertices are partitioned in to K>=2 disjoint sets Vi,1<=i<=k.The multi stage graph problem is to find a minimum cost paths from s(source) to t(sink)
Two approach(forward and backward)

8. Define All pair shortest path problem

Given a weighted connected graph, all pair shortest path problem asks to find the lengths of shortest paths from each vertex to all other vertices.

9.Define Distance matrix

Recording the lengths of shortest path in n x n matrix is called Distance matrix(D)

10.Define floyd's algorithm

- ⇨ To find all pair shortest path
- ⇨ The algorithm computes the distance matrix of a weighted graph with n vertices through series of n by n matrices :D(0)…D(k-1),D(k)…..D(n)

11. State the time efficiency of floyd's algorithm

$O(n^3)$

It is cubic

12. Define OBST

- ⇨ Dynammic pgmg. Used
- ⇨ If probabilities of searching for elements of a set are known then finding optimal BST for which the average number of comparisons in a search is smallest possible.

13. Define catalan number

The total number of binary search trees with n keys is equal to nth catalan number

C(n)=(2n to n) 1/(n+1) for n>0,c(0)=1

14. State time and space efficiency of OBST

SPACE EFFICIENCY :QUADRATIC
TIME EFFICIENCY : CUBIC.

UNIT 4 AND 5

UNIT IV BACKTRACKING

The General method- Solution space and tree organization- The Eight Queens problem- Sum of subset problem- Graph coloring- Knapsack problem

UNIT V BRANCH AND BOUND

The General method- O/I Knapsack problem- Traveling sales person problem-Efficiency consideration . NP Hard and NP Complete problems - Basic concepts

1. What are the requirements that are needed for performing Backtracking?

To solve any problem using backtracking, it requires that all the solutions satisfy a complex set of constraints. They are:

i. Explicit constraints.

ii. Implicit constraints.

2.Define explicit constraint.

They are rules that restrict each x_i to take on values only from a give set. They depend on the particular instance I of the problem being solved. All tuples that satisfy the explicit constraints define a possible solution space.

3. Define implicit constraint.

They are rules that determine which of the tuples in the solution space of I satisfy the criteria function. It describes the way in which the x_i must relate to each other.

4.Define state space tree.

The tree organization of the solution space is referred to as state space tree.

5.Define state space of the problem.

All the paths from the root of the organization tree to all the nodes is called as state space of the problem

6.Define answer states.

Answer states are those solution states s for which the path from the root to s defines a tuple that is a member of the set of solutions of the problem.

7.What are static trees?

The tree organizations that are independent of the problem instance being solved are called as static tree.

8.What are dynamic trees?

The tree organizations those are independent of the problem instance being solved are called as static tree.

9.Define a live node.

A node which has been generated and all of whose children have not yet been generated is called as a live node.

10. Define a E – node.

E – node (or) node being expanded. Any live node whose children are currently being generated is called as a E – node.

11.Define a dead node.

Dead node is defined as a generated node, which is to be expanded further all of whose children have been generated.

12.,What are the factors that influence the efficiency of the backtracking algorithm?

The efficiency of the backtracking algorithm depends on the following four factors. They are:

i. The time needed to generate the next x_k
ii. The number of x_k satisfying the explicit constraints.
iii. The time for the bounding functions B_k
iv. The number of x_k satisfying the $B_{k.}$

13.Define Branch-and-Bound method.

The term Branch-and-Bound refers to all the state space methods in which all children of the E-node are generated before any other live node can become the E- node.

14.What are the searching techniques that are commonly used in Branch-and-Bound method.

The searching techniques that are commonly used in Branch-and-Bound method are:

i. FIFO
ii. LIFO
iii. LC
iv. Heuristic search

15.State 8 – Queens problem.

The problem is to place eight queens on a 8 x 8 chessboard so that no two queen "attack" that is, so that no two of them are on the same row, column or on the diagonal.

16.State Sum of Subsets problem.

Given n distinct positive numbers usually called as weights , the problem calls for finding all the combinations of these numbers whose sums are m.

17. State m – colorability decision problem.

Let G be a graph and m be a given positive integer. We want to discover whether the nodes of G can be colored in such a way that no two adjacent nodes have the same color yet only m colors are used.

18.Define chromatic number of the graph.

The m – colorability optimization problem asks for the smallest integer m for which the graph G can be colored. This integer is referred to as the chromatic number of the graph.

19. Define a planar graph.

A graph is said to be planar iff it can be drawn in such a way that no two edges cross each other.

20. What are NP- hard and Np-complete problems?

The problems whose solutions have computing times are bounded by polynomials of small degree.

21. What is a decision problem?

Any problem for which the answer is either zero or one is called decision problem.

22. what is approximate solution?

A feasible solution with value close to the value of an optimal solution is called approximate solution.

23. what is promising and non-promising nodes?

a node in a state space tree is said to be promising if it corresponds to a partially constructed solution from which a complete solution can be obtained.

The nodes which are not promising for solution in a state space tree are called

non-promising nodes.

24.Write formula for bounding function in Knapsack problem

In knapsack problem upper bound value is computed by the formula

UB = v + (W-w) * (vi+1/wi+1)

25. **Write about traveling salesperson problem.**

Let g = (V, E) be a directed. The tour of G is a directed simple cycle that includes every vertex in V. The cost of a tour is the sum of the cost of the edges on the tour. The traveling salesperson problem is to find a tour of minimum cost.

In branch and bound technique of TSP problem Lower bound lb= s/2 ⌈ ⌉

26. Write some applications of traveling salesperson problem.

-> Routing a postal van to pick up mail from boxes located at n different sites.
-> Using a robot arm to tighten the nuts on some piece of machinery on an assembly line.
-> Production environment in which several commodities are manufactured on the same set of machines.

27. Give the time complexity and space complexity of traveling salesperson problem.

Time complexity is O ($n^2 2^n$). Space complexity is O ($n 2^n$).

28. Differentiate decision problem and optimization problem

Any problem for which the answer is either zero or one is called decision problem

Any problem that involves the identification of an optimal (maximum or minimum) value of a given cost function is called optimization problem

29. what is class P and NP?

P is set of all decision problems solvable by deterministic algorithms in polynomial time.

NP is set of all decision problems solvable by non deterministic algorithms in polynomial time.

30. Define NP-Hard and NP-Complete problems

Problem L is NP-Hard if and only if satisfiability reduces to L.

A Problem L is NP-Complete if and only if L is NP-Hard and L belongs to NP.