Questions/Answer in this PDF

1. **Explain How backtracking is used to solve 8 queens problem?**
2. **Explain BFS with the help of Example**
3. **Explain DFS with the help of Example**
4. **Give applications of BFS and DFS.**
5. **Explain P,NP, NP Hard and NP Complete with the help of example.**
6. **Explain Naïve String matching method.**
7. **Explain Hamiltonian Cycle.**

Que 14 Explain how backtracking is used to solve the 8-queens problem.

Ans. The basic idea of backtracking is to build up a vector, one component at a time and to test whether the vector being formed has any chance of success.

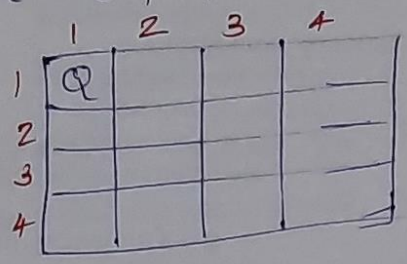
- The major advantage of this method is that we can realize the fact that the partial vector generated does not lead to an optimal solution.
- In such a situation that vector can be ignored.
- Backtracking algorithm determines the solution by systematically searching the set of all feasible solutions for the given problem.

Now, The n-queens problem can be stated as follows:

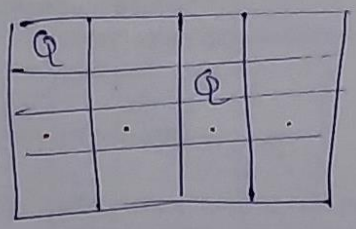
→ "Consider a nxn chessboard on which we have to place n queens so that no two queens attack each other by being in the same row or in the same column or on the same diagonal.

- first we will see for 4-queens.
- we start with empty chessboard.
- Then we place queen 1 in the first possible position of its row.

  i.e. on 1st row and 1st column.

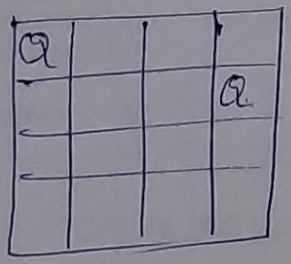


- Then place queen 2 after trying unsuccessful place (1,2), (2,1), (2,2). at position (2,3).

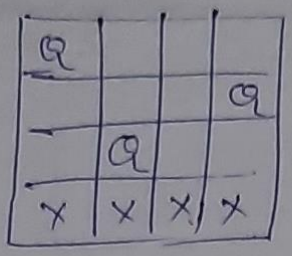


→ This is the dead end because 3rd queen can not be placed in next column, as there is no acceptable position for queen 3.

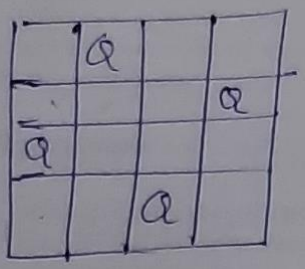
→ Hence algorithm backtracks and places 2nd queen at (2,4) position.



- The place 3rd queen at (3,2) but it is again another dead end as next queen (4th queen) can not be placed at permissible position.



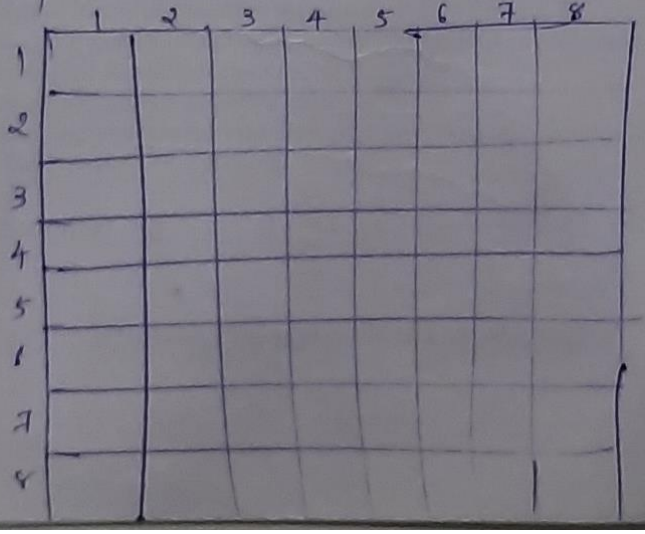
- Hence we need to backtrack all the way upto queen 1 and move it to (1,2).
- Repeat the same procedure and place queen 1 at (1,2), queen2 at (2,4), queen3 at (3,1) and queen4 at (4,3).



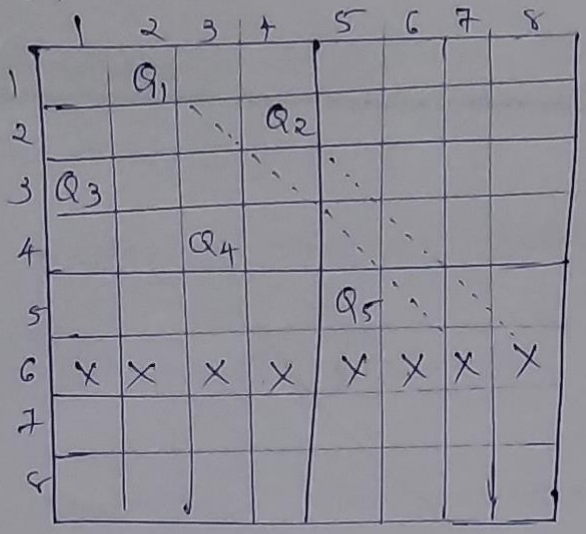
Thus solution is obtained for 4 queens.

⇒ Now we will consider how to place 8 queens on the chessboard.

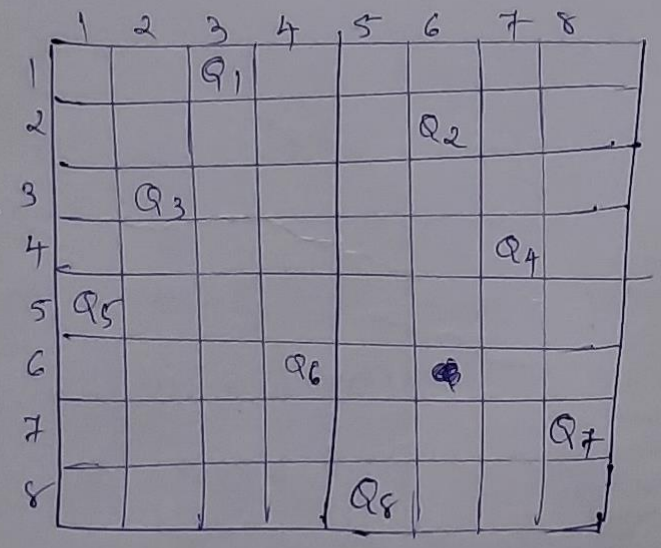
- Initially chessboard is empty.



- Now we will consider start placing the queens on the chessboard.



- Now if we tries to place Q6 then it will not be possible to put it in a chessboard.
- So we need to backtrack.
- Similar way we can check for each and every possibility and final we will obtain solution.



- Finally the successful placement of all the eight queens is shown in above figure.

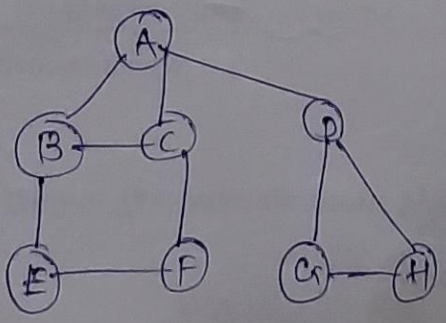
Que 11 Explain Breadth first search algorithm with the help of example. (BFS)

Ans: Searching a graph means traversing a graph from given vertex.

→ The breadth first search follows the following rule.

1. Select an unvisited node x, visit it, have it be the root in a BFS tree being formed. Its level is called the current level.
2. From each node x in the current level, in the order in which the level nodes were visited, visit all the unvisited neighbours of x.
   - The newly visited nodes from this level form a new level. This new level becomes the next current level.
3. Repeat step 2 for all the unvisited vertices.
4. Repeat from step 1 until no more vertices are remaining.

→ Now Consider the given graph.

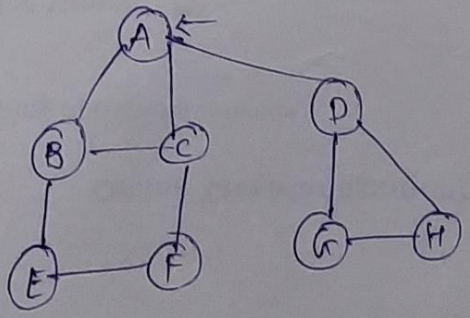


```
Algorithm BFS(G)
{
// Problem Description: This algorithm is for finding BFS.
   Queue Q
   while (G has unvisited node) do
   {
        V ← an unvisited node;
        visit[V] ← 1;
        en_que(V, Q)
        while (Q is not empty) do
        {
            X ← del_que(Q)
            for (unvisited neighbour Y of X) do
            {
                visit[Y] ← 1
                en_que(Y, Q)
            }
        }
   }
}
```

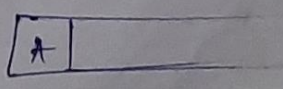
Now we will generate the bfs tree for the given graph.

- BFS uses for queue data structure.

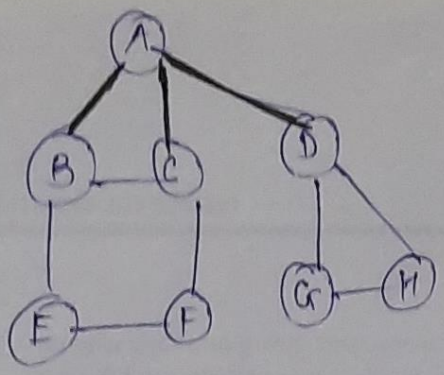
Step 1:



we will start search from node A and insert it into queue.



Step 2:

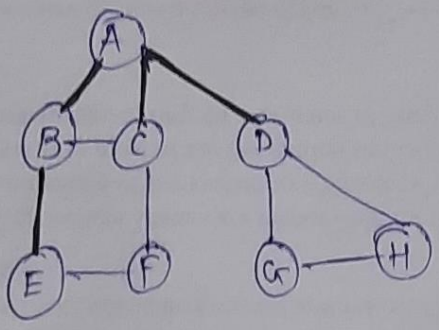


Adjacent nodes to A will be obtained and inserted in the queue. delete A from queue and print it.

B	C	D.

A - deleted nodes.

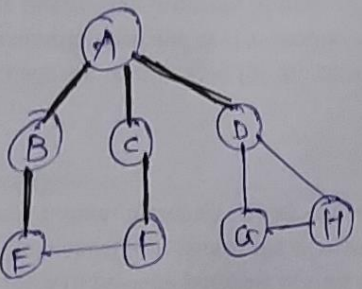
Step 3:



C	D	E

A, B - deleted nodes.

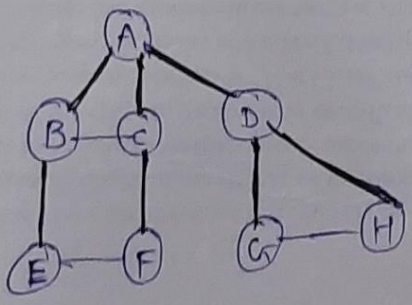
Step 4:



D	E	F

A, B, C - deleted nodes.

Step 5:

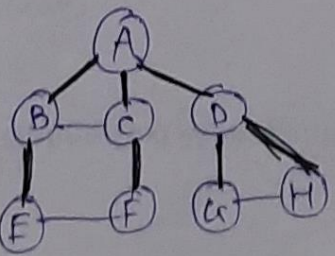


D	E	F	G

E	F	G	H

A, B, C, D - deleted nodes.

Step 6:



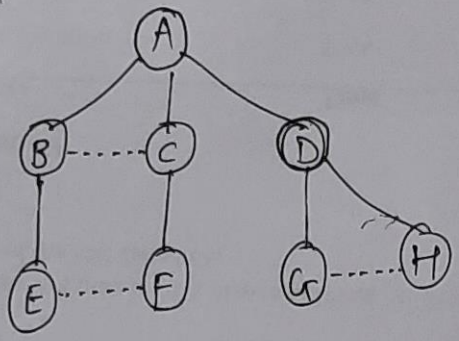
- there is no unvisited node remaining. we will delete the vertices from queue and print.

So, A, B, C, D, E, F, G, H

Tree edge:- In a graph G containing an edge (u, v) if a new unvisited vertex v is reached from the current vertex then edge (u, v) is called tree edge.

Cross edge:- If an edge leading to its previously visited vertex other than its immediate predecessor is encountered then that edge is called cross edge.

→ The cross edge are shown by dotted line and tree edge are shown by solid edge.



Que 12 Explain Depth first search algorithm with the help of example. (DFS)

Ans: Searching a graph means traversing a graph from given vertex.

- DFS follows the following rules:

1. Select an unvisited node v, visit it, and treat as the current node.
2. Find an unvisited neighbours of the current node, visit it, and make it the new current node.

3. If the current node has no unvisited neighbours, backtrack to its parent, and make it new current node.

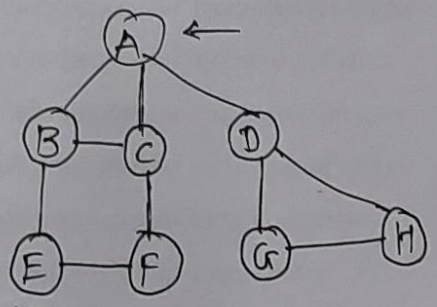
4. Repeat the step 2 and 3 until no more nodes can be visited.

5. Repeat from step 1 for the remaining nodes.

Algorithm DFS(V)

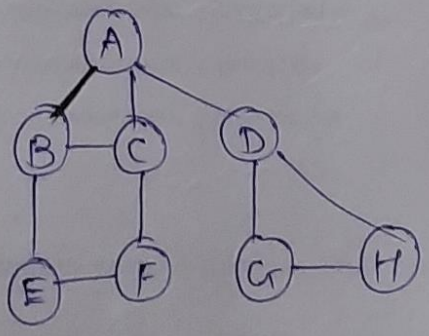
```
{
    visit[V] = 1
    for (each vertex x adjacent from V)
    {
        if (visit[X] = 0) then
            DFS(X)
    }
}
```

Example:

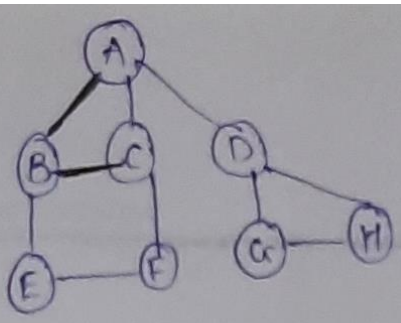


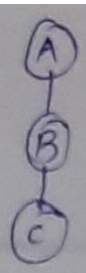
we will start search from node A.

Step 2:

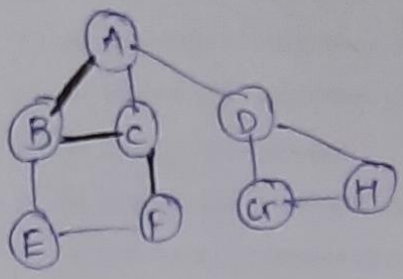


Step 3:

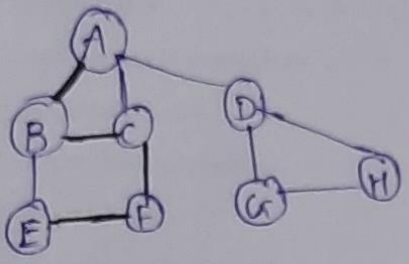


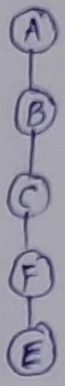


Step 4:

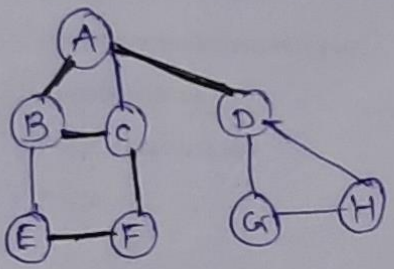


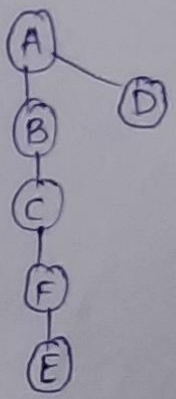
Step 5: there is unvisited node E. visit it.



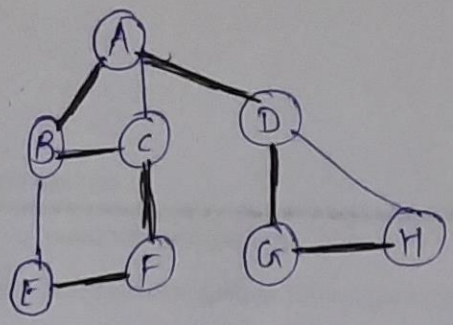


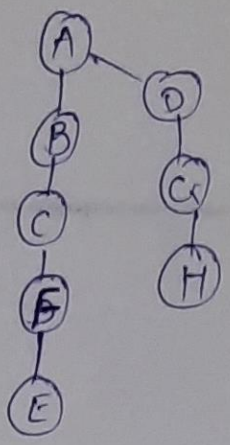
Step 6: Now there is no reachable nodes from E. so we will backtrack and proceed.





Step 7: Same way we will visit G and then H.



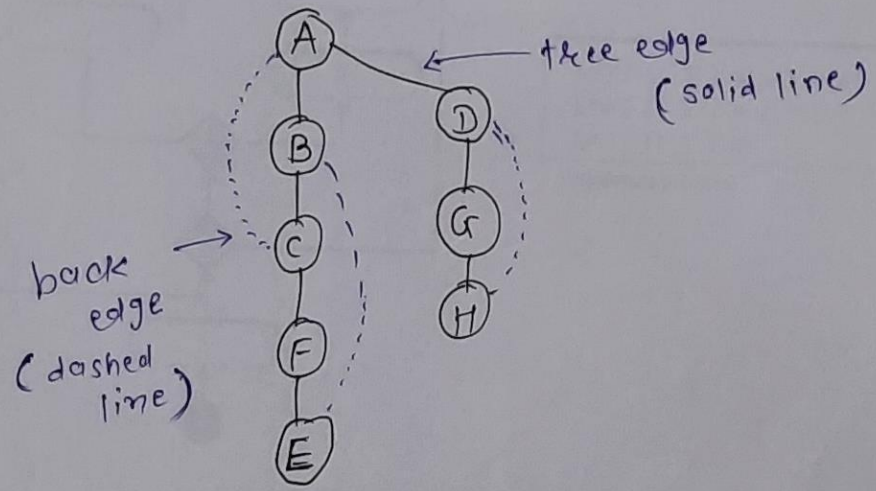


Hence we obtained DFS sequence as

A, B, C, F, E, D, G, H.

Tree edge:- In a graph G containing an edge (u,v) if a new unvisited vertex v is reached from the current vertex then edge (u,v) is called tree edge.

Back edge:- In a graph G if a previously visited vertex v is reached from the current vertex u then edge (u,v) is called back edge.



Que 13 Give the application of BFS and DFS.

Ans: ⇒ Application of Breadth first search (BFS):

1. For finding the connected components in the graph.
2. For checking if any cycle exists in the given graph.
3. To obtain shortest path between two vertices.

⇒ Application of Depth First search:

1. DFS is used for checking connectivity of a graph.
   - Start traversing the graph using depth first method and after an algorithm halts if all the vertices of graph are visited then the graph is said to be a connected graph.

2. DFS is used for checking acyclicity of graph. If the DFS does not have back edge then the graph is said to be acyclic (not cycle exists).

3. DFS is used to find an articulation point.

Question: Explain P,NP, NP Hard and NP Complete with the help of example.

P Class

The P in the P class stands for Polynomial Time. It is the collection of decision problems (problems with a "yes" or "no" answer) that can be solved by a deterministic machine in polynomial time.

Features:

1. The solution to P problems is easy to find.
2. P is often a class of computational problems that are solvable and tractable. Tractable means that the problems can be solved in theory as well as in practice. But the problems that can be solved in theory but not in practice are known as intractable.

Example:

- Searching of an element
- Inserting an element in linked list
- Sorting data using selection sort
- Sorting data using merge sort

NP Class

The NP in NP class stands for Non-deterministic Polynomial Time. It is the collection of decision problems that can be solved by a non-deterministic machine in polynomial time.

Features:

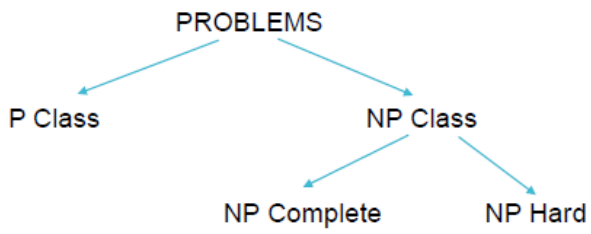
The solutions of the NP class are hard to find since they are being solved by a non-deterministic machine but the solutions are easy to verify.

Problems of NP can be verified by a Turing machine in polynomial time.

Examples :

Knapsack problem

Travelling Salesman problem



NP Complete Problems

Decision problem P is called NP complete if it has following two properties:

1. It belongs to class NP
2. Every other problem in NP can be transformed to P in polynomial time.

Examples:
Knapsack problem
Travelling salesman problem
Hamiltonian path problem

NP Hard Problems

A Problem X is NP-Hard if there is an NP-Complete problem Y, such that Y is reducible to X in polynomial time. NP-Hard problems are as hard as NP-Complete problems. NP-Hard Problem need not be in NP class.

To solve this problem, it do not have to be in NP .

Do not have to be a Decision problem.

Example: Halting problem, Vertex cover problem, etc.

String Matching

- String matching algorithm are normally used in text processing
- String matching means finding one or more generally all occurrences of a string in the text.
- These occurrences are called pattern.
- Hence, sometimes string matching algo are also called pattern matching algorithms.

Text T is denoted by

$t_0 \ldots t_{n-1}$ and

Pattern P is denoted by

$p_0 \ldots p_{m-1}$

⇒ Three types:

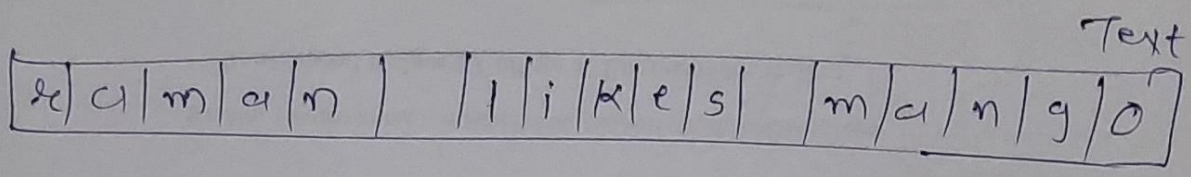
1. The naive method,
2. Rabin-Karp method,
3. Finite automaton for string matching

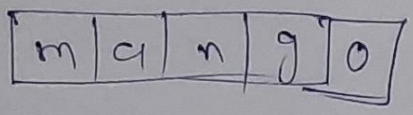
1. The naive method:-

This is the simplest method which works using Brute force approach.

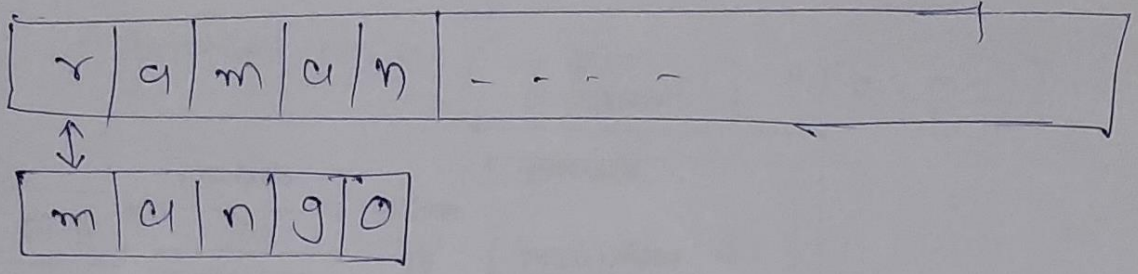
- It is a straight forward approach for solving the problem.
- This method has a "just do it" approach.
- This algorithm performs a checking at all positions in the text between 0 to n-m, whether an occurrence of the pattern starts there or not.
- Then after each attempt, it shifts the pattern by exactly one position to the right.
- If the match is found then it returns otherwise the matching process is continued by shifting one character to the right.

* Consider example:

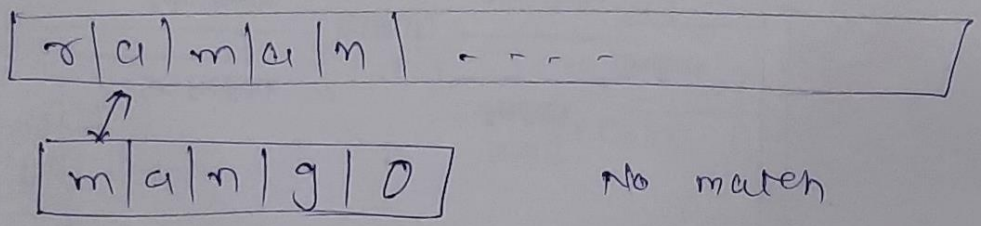




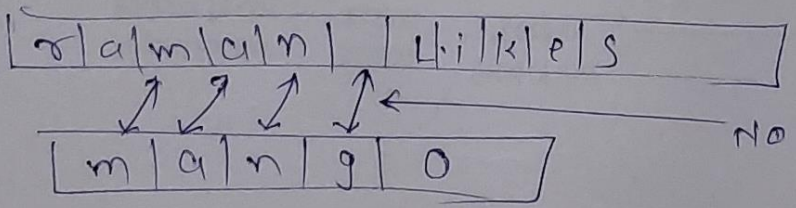
Pattern



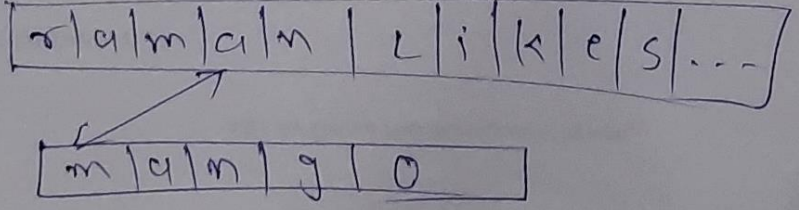
No match found ∴ Shift to the right by 1 position.

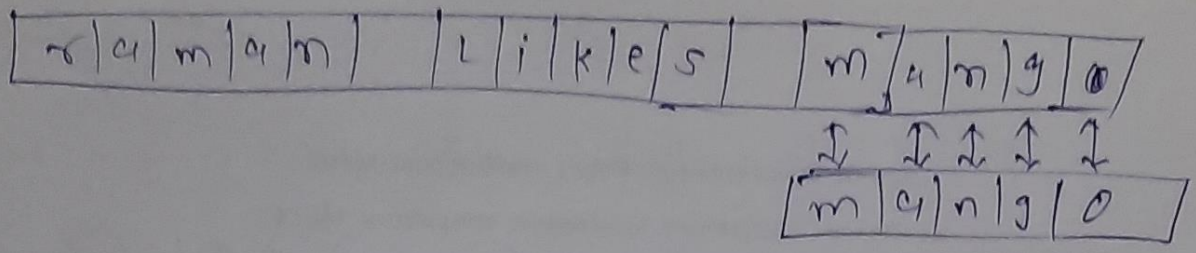


No match



No match,





The match is found.

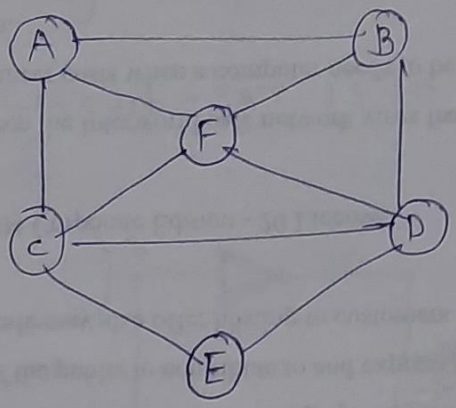
Algorithm:

```
Algorithm Naive ( T[0..n-1], P[0...m-1])
{
    for i ← 0 to n-m do
    {   j ← 0
        while j < m and P[j] = T[i+j] do
            j ← j+1
        if (j = m)
            return i
    }
    return -1
}
```

Que: Hamiltonian cycle :-

Ans: This is a problem in which graph G is accepted as input and it is asked to find a simple cycle in G that visits each vertex of G exactly ones and returns to its starting vertex.

- Such a cycle is called Hamiltonian cycle.



A - B - D - E - C - F - A