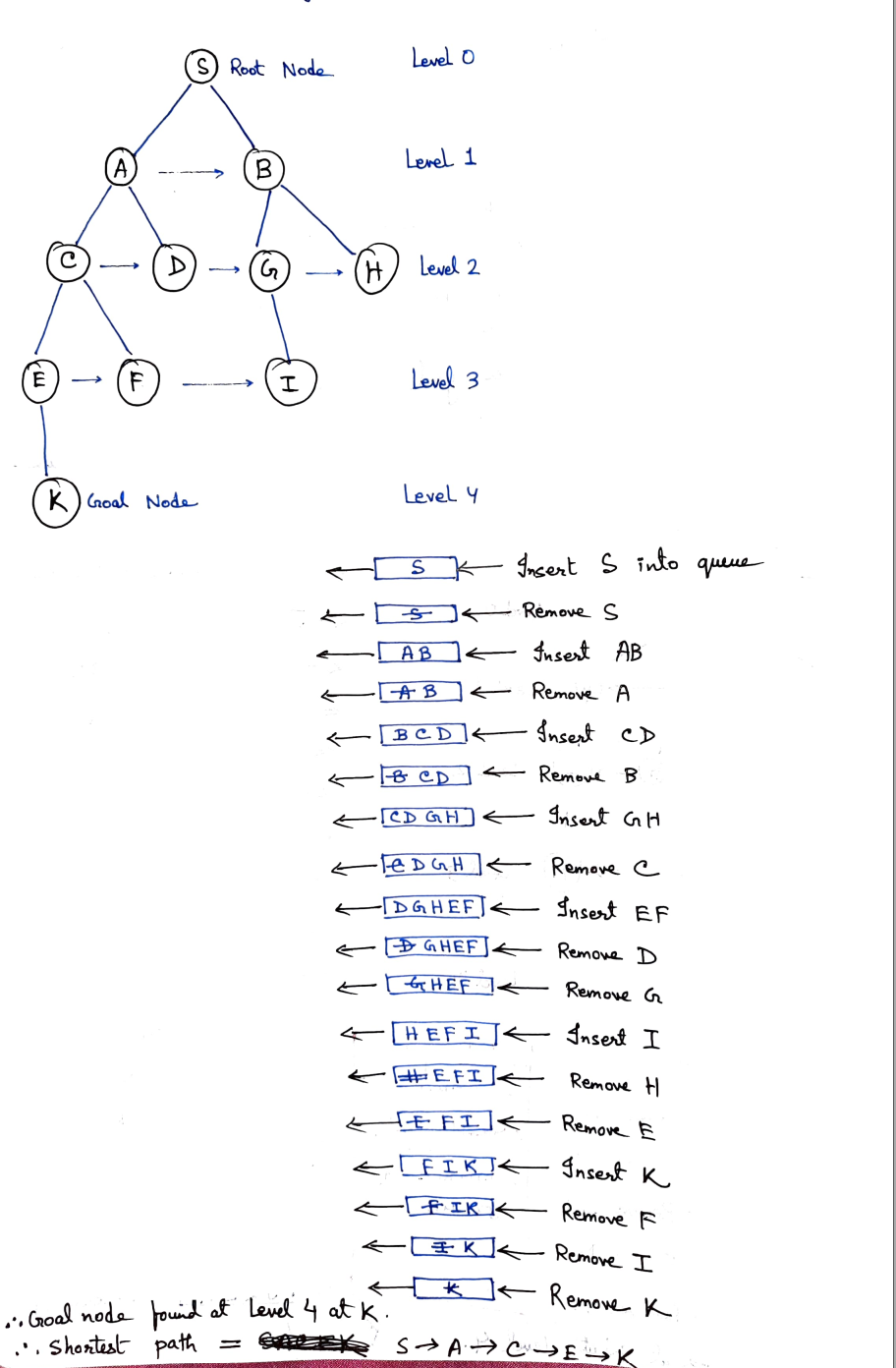
***Graph Traversal in Artificial Intelligence***

**Breadth First Search(BFS):** Breadth First Search algorithm is a graph traversing technique, where you select a random initial node (source or root node) and start traversing the graph layer-wise in such a way that all the nodes and their respective children nodes are visited and explored.

Here‘s the pseudocode to implement the Breadth First Search Algorithm:

1. Input: s as the source node
2. BFS (G, s)
3. let Q be queue.
4. Q.enqueue( s )
5. mark s as visited
6. while ( Q is not empty)
7. v = Q.dequeue( )
8. for all neighbors w of v in Graph G
9. if w is not visited
   1. Q.enqueue( w )
   2. mark w as visited



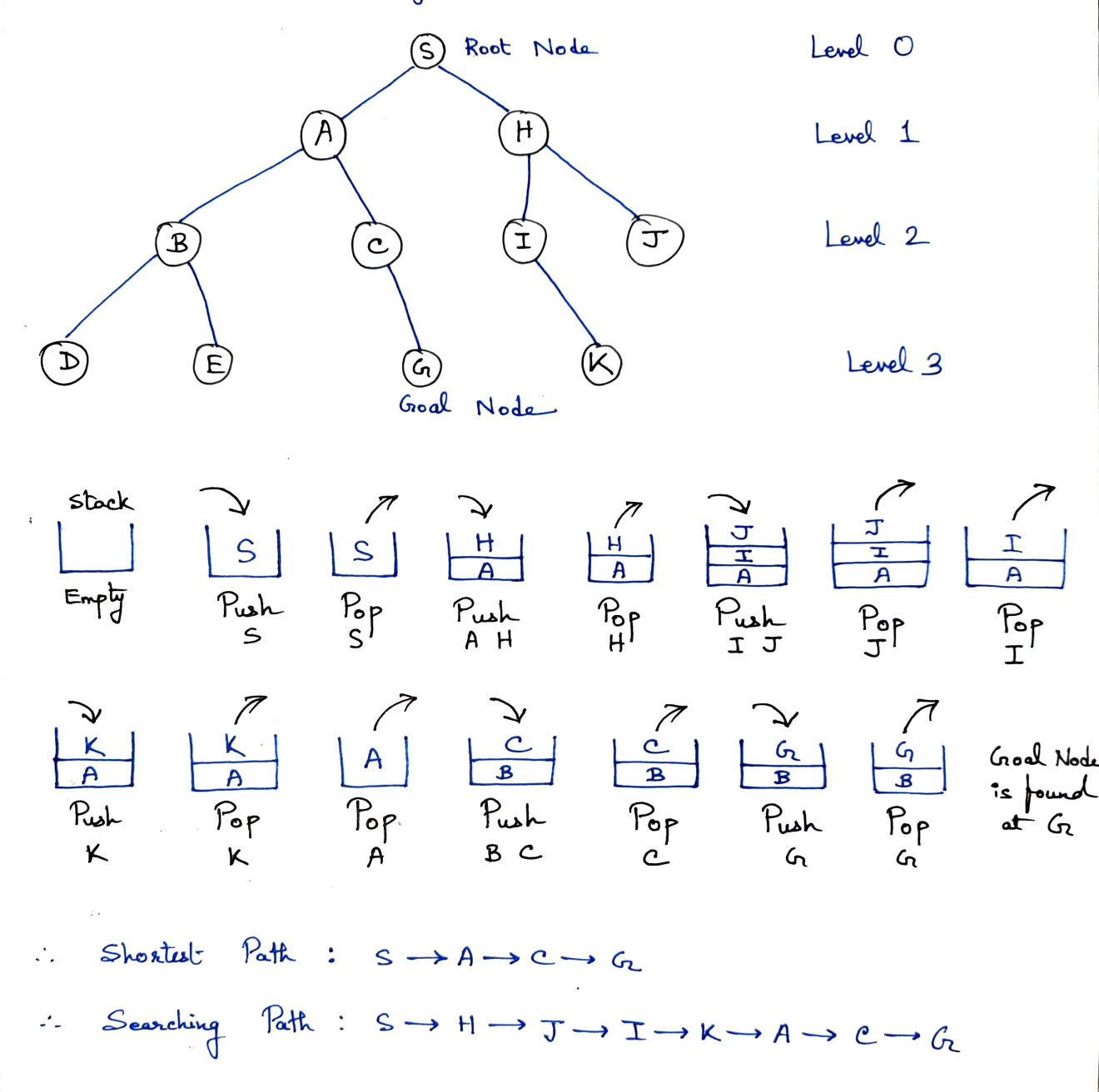
**Applications Of BFS Algorithm:**

* Crawlers in Search Engines
* GPS Navigation systems
* Find the Shortest Path & Minimum Spanning Tree for an unweighted graph
* Broadcasting

**Depth First Search(DFS):** Depth-first search (DFS) is an algorithm for traversing or searching tree or graph data structures. The algorithm starts at the root node (selecting some arbitrary node as the root node in the case of a graph) and explores as far as possible along each branch before backtracking.

Here‘s the pseudocode to implement the Depth First Search Algorithm:

1. SET STATUS = 1 (ready state) for each node in G
2. Push the starting node A on the stack and set its STATUS = 2 (waiting state)
3. Repeat Steps 4 and 5 until STACK is empty
4. Pop the top node N. Process it and set its STATUS = 3 (processed state)
5. Push on the stack all the neighbours of N that are in the ready state (whose STATUS = 1) and set their  
   STATUS = 2 (waiting state)  
   [END OF LOOP]
6. EXIT

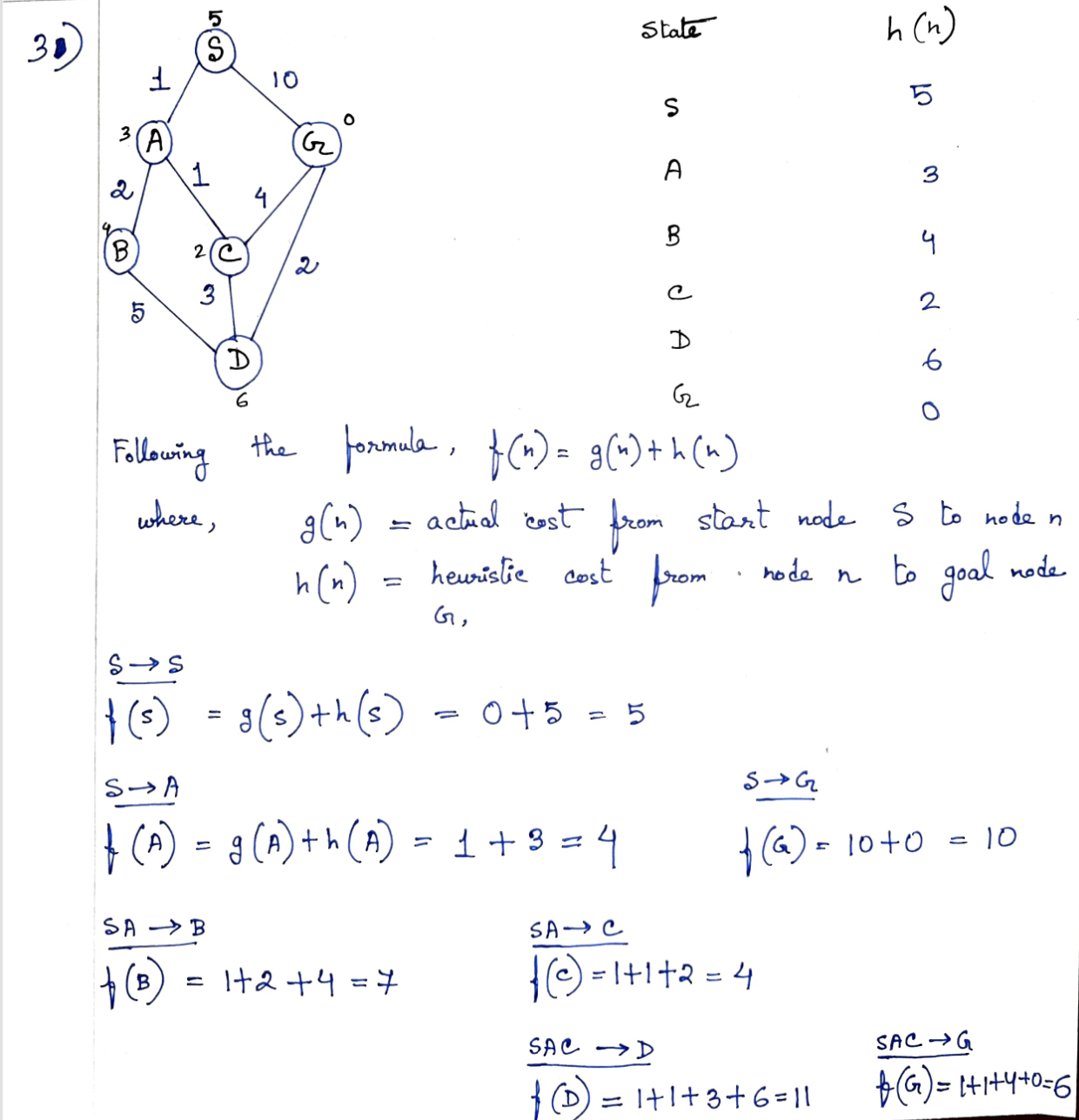


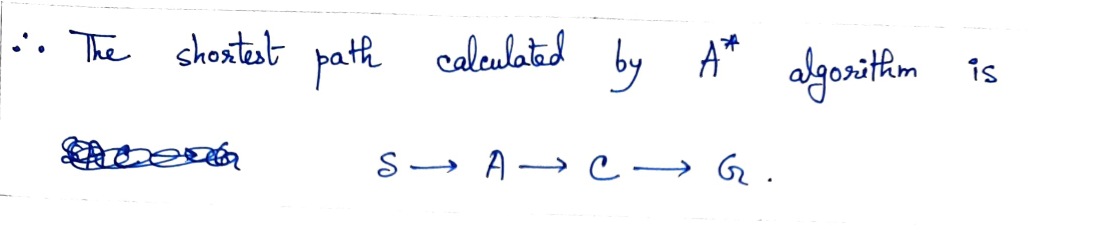
**Applications Of DFS Algorithm:**

* If we perform DFS on unweighted graph, then it will create minimum spanning tree for all pair shortest path tree
* We can detect cycles in a graph using DFS. If we get one back-edge during BFS, then there must be one cycle.
* Using DFS we can find path between two given vertices u and v.
* We can perform topological sorting is used to scheduling jobs from given dependencies among jobs. Topological sorting can be done using DFS algorithm.
* Using DFS, we can find strongly connected components of a graph. If there is a path from each vertex to every other vertex, that is strongly connected.

**A\* Algorithm:** A\* Algorithm is one of the best and popular techniques used for path finding and graph traversals.

* It maintains a tree of paths originating at the start node.
* It extends those paths one edge at a time.
* It continues until its termination criterion is satisfied.
* A\* Algorithm extends the path that minimizes the following function-
* **f(n) = g(n) + h(n)**
* Here,
* ‘n’ is the last node on the path
* g(n) is the cost of the path from start node to node ‘n’
* h(n) is a heuristic function that estimates cost of the cheapest path from node ‘n’ to the goal node





A\* is often used for the common [pathfinding](https://en.wikipedia.org/wiki/Pathfinding) problem in applications such as video games, but was originally designed as a general graph traversal algorithm. It finds applications in diverse problems, including the problem of [parsing](https://en.wikipedia.org/wiki/Parsing) using [stochastic grammars](https://en.wikipedia.org/wiki/Stochastic_context-free_grammar) in [NLP](https://en.wikipedia.org/wiki/Natural_language_processing). Other cases include an Informational search with online learning.