**Explanation:**

### ****1. DFS in Tree****

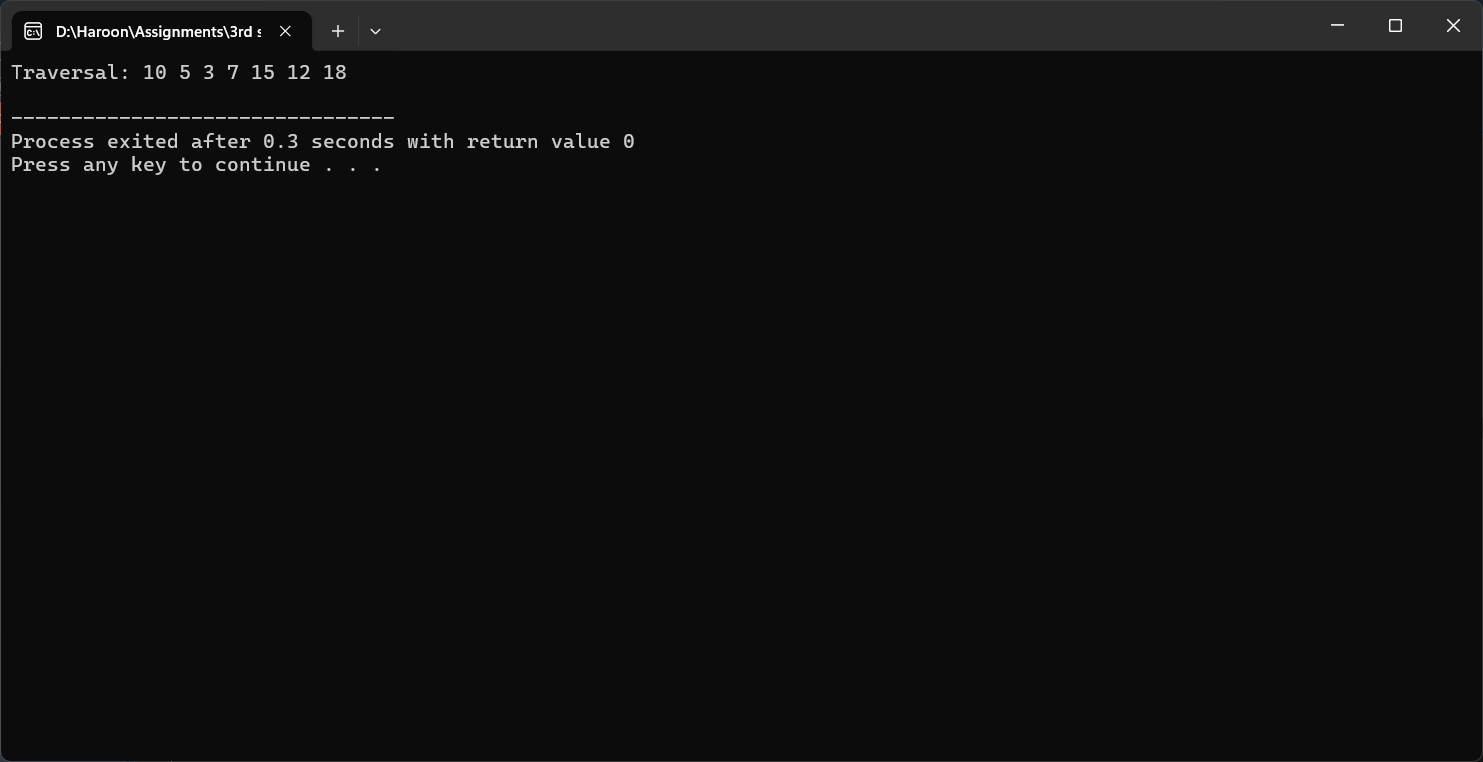
DFS (Depth First Search) is a tree traversal technique where:

Start at the root node.

Visit nodes by going as deep as possible along each branch before backtracking.

Implemented using recursion.  
DFS helps explore all nodes and is useful for tasks like searching and pathfinding.

**Output:**



### ****2. DFS in Graph****

DFS in graphs works similarly to trees but:

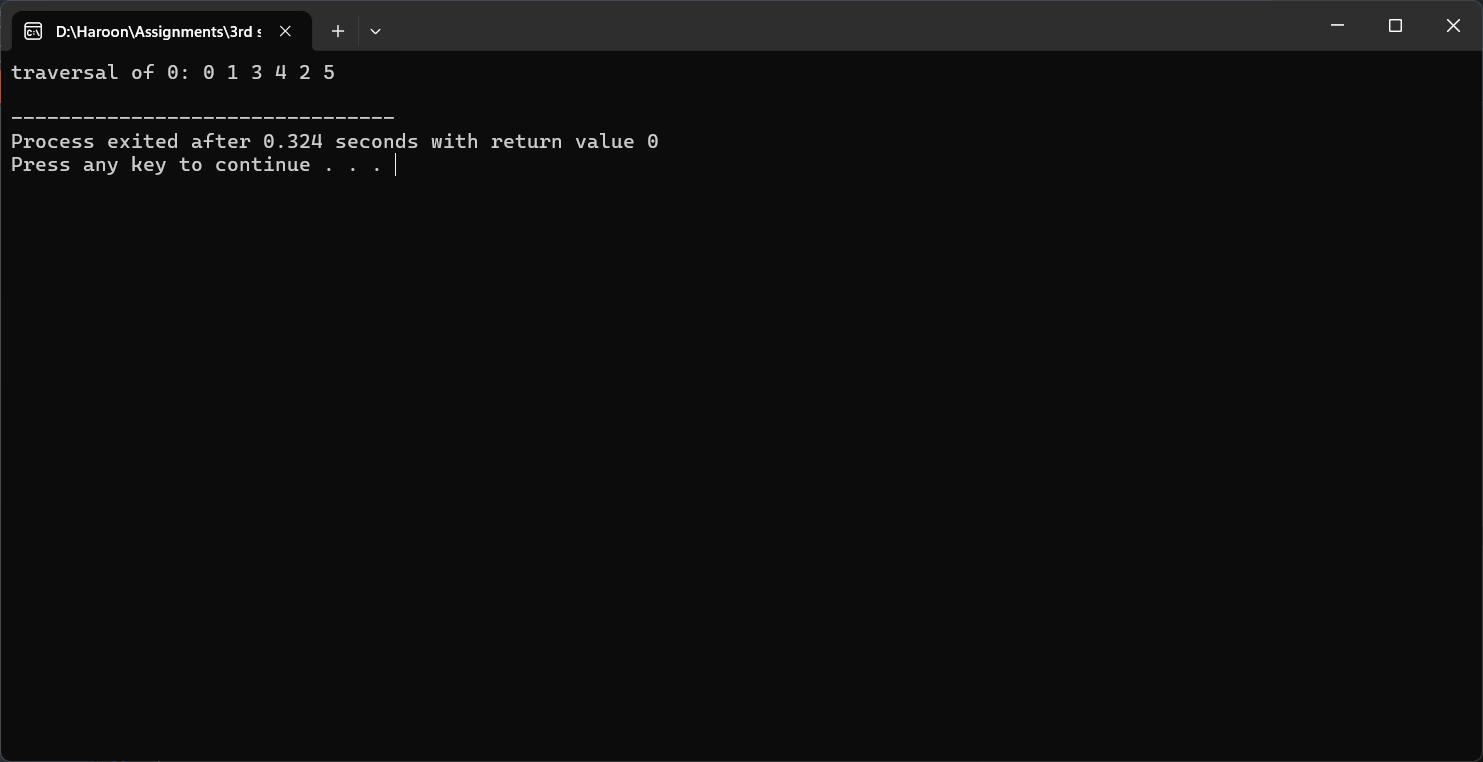
It needs a visited array to avoid cycles.

Visits one node, then recursively visits all unvisited neighbors.

Works for both directed and undirected graphs.

DFS is used in cycle detection, pathfinding, and connectivity checking.

**Output:**



### ****3. BFS in Tree****

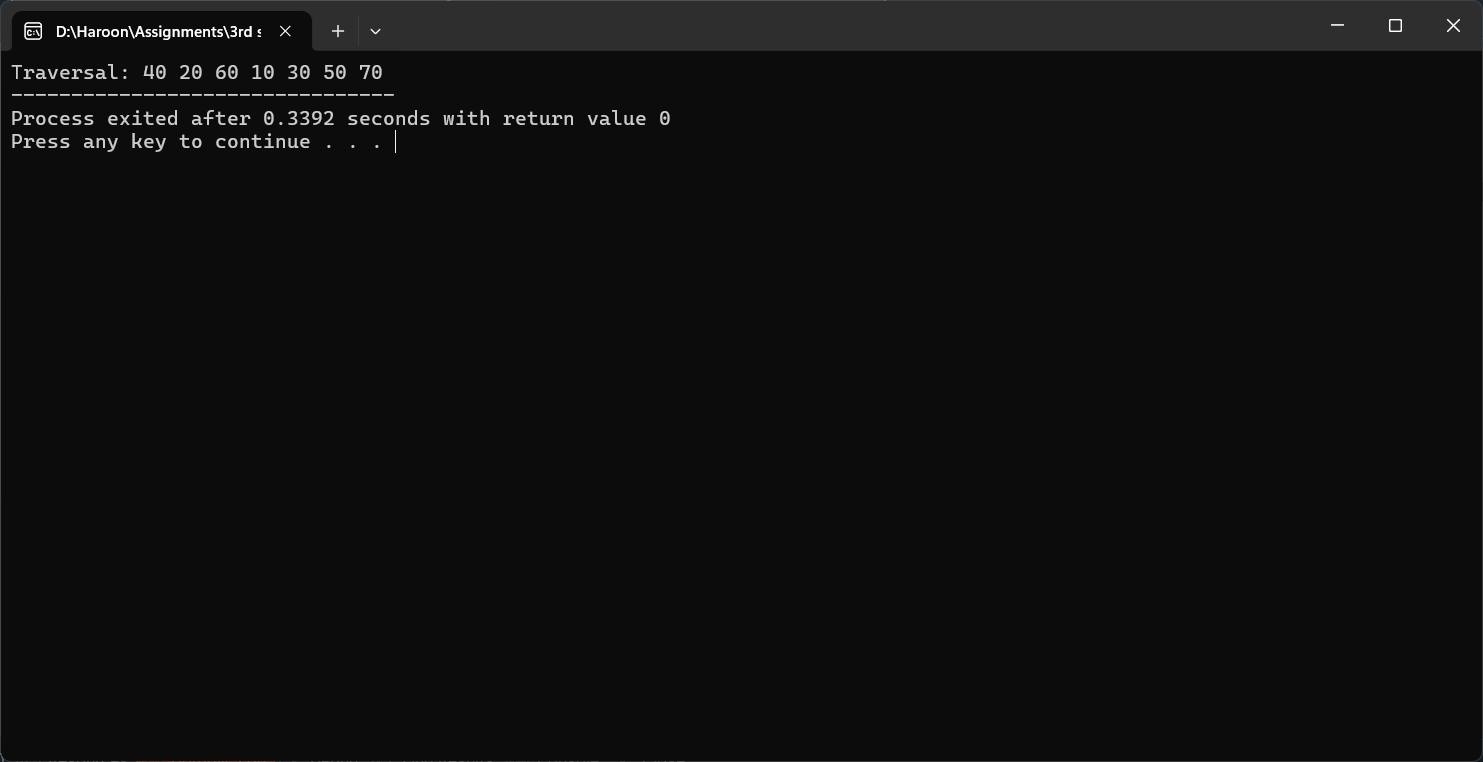
BFS (Breadth First Search) is a level-by-level traversal:

Starts at the root and visits all nodes at the current level before going deeper.

Uses a queue (custom queue here) to keep track of nodes.

Useful for finding shortest paths or exploring all levels.

**Output:**



### ****4. BFS in Graph****

BFS in graphs also uses a queue:

Starts at a node and explores its neighbors first.

Keeps track of visited nodes to prevent loops.

Used for shortest path in unweighted graphs, and level order traversal.

**Output:**

