task 1(a): The code reads an ~~output~~ input file where it makes a adjacency matrix of the graph.

task1(b): The code reads an input file where it makes a adjacency list of the graph which shows the graph node's relation.

task 2: The code reads a input file where it creats adjacency ~~list~~ matrix, list and peteforms a bfs traversal and gives a output.

task 3: The code reads a input file where it creats a adjacency matrix list and performs a DFS traversal.

task 4: The code reads an input from the code and performs a BFS/DFS to find weither there is a cycle or not. Basically the code is about cycle finding code.

task 5: The code uses BFS to find the shortest distance and path from the root node (node 1) to a given destination node in an undirected graph represented as an adjacency matrix. It stores visited nodes, distances and predecessor information during the BFS traversal. the shortest distance and path are are pointed and written to an output file. If the distination node is unreachable, it outputs a path with only the destination node and a large distance of 1000.

task 6: The code uses a flood-fill algorith (DFS) to find the ~~min~~ maximum count of non water cells in the given grid represented by a 2D matrix. It explores all non-visited cells that are not water cells "0" and counts the connected non-water cells each region. The maximum count is then printed and written to the output file.