Task 01 (a).

- created adjacency matrix by initializing nxn matrix setting all vals to zero
- handled the 'index 0' vertex on function call by adding up another vertix. associated with weight 0.

Task 01 (b)

- adjacency list was created by initializing a list of range(n)
- appended each vertix and weight to corresponding edges.

Task 02:

- Emaded BFS from pseudocode for graph representation (using BFS traversal)
- Output as adjacency list; which was done using BFS traversal starting from 1st Node

Task 03:

- Handled multiple input files
- DFS implementation
- initialized empty graph with (N+1) vertices
- DFS algo explores all the vertices & creates DFS ~~grap~~ path.

Task 04

- FindMyCycle implements DFS
- is_cycle checks & calls Find My. cycle by finding cycle returns Yes/No
- My code Handles multiple input files with an additional function.

Task 05

- Shortest path calc was done by using DFS on undirected graph
- stored graph in dict

  keys = city
  values = list of neighbors

Task 06

- Counts diamond using DFS.
- starting point (n, c
- For each unvisited calls DFS; Also checks & counts diamonds +
- Then updates diamond count