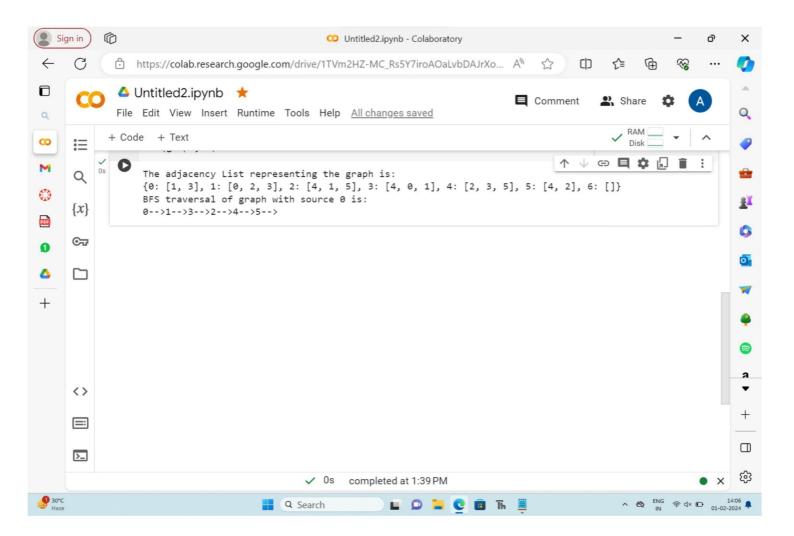
Implement Breadth First Search Algorithm using a Queue

```
from queue import Queue
graph = {0: [1, 3], 1: [0, 2, 3], 2: [4, 1, 5], 3:
[4, 0, 1], 4: [2, 3, 5], 5: [4, 2], 6: []}
print("The adjacency List representing
the graph is:")
print(graph)
def bfs(graph, source):
  Q = Queue()
  visited_vertices = set()
  Q.put(source)
  visited_vertices.update({0})
  while not Q.empty():
     vertex = Q.get()
     print(vertex, end="-->")
     for u in graph[vertex]:
       if u not in visited_vertices:
          Q.put(u)
          visited_vertices.update({u})
print("BFS traversal of graph with source
0 is:")
bfs(graph, 0)
```



Implement Depth First Search Algorithm

```
code:graph1 = {
  'A': ['B','S'],
  'B': ['A'],
  'C':['D','E','F','S'],
  'D': ['C'],
  'E':['C','H'],
  'F': ['C','G'],
  'G': ['F','S'],
  'H': ['E','G'],
  'S': ['A','C','G']
}
def dfs(graph, node, visited):
  if node not in visited:
     visited.append(node)
     for k in graph[node]:
        dfs(graph,k, visited)
  return visited
visited = dfs(graph1,'D', [])
print(visited)
```

