Problem – 2 : Given an undirected graph, return a vector of all nodes by traversing the graph using breadth-first search (BFS).

```
from collections import deque

def bfs(graph, start_node):
    visited = set()
    queue = deque([start_node])
    visited.add(start_node)
    result = []

    while queue:
        current_node = queue.popleft()
        result.append(current_node)

        for neighbor in graph[current_node]:
            if neighbor not in visited:
                visited.add(neighbor)
                queue.append(neighbor)

    return result

graph = {
    'A': ['B', 'C'],
    'B': ['A', 'D', 'E'],
    'C': ['A', 'F', 'G'],
    'D': ['B'],
    'E': ['B'],
    'F': ['C'],
    'G': ['C']
}
```

```
start_node = 'A'

bfs_result = bfs(graph, start_node)

print("BFS traversal starting from node {}: {}".format(start_node, bfs_result))
```