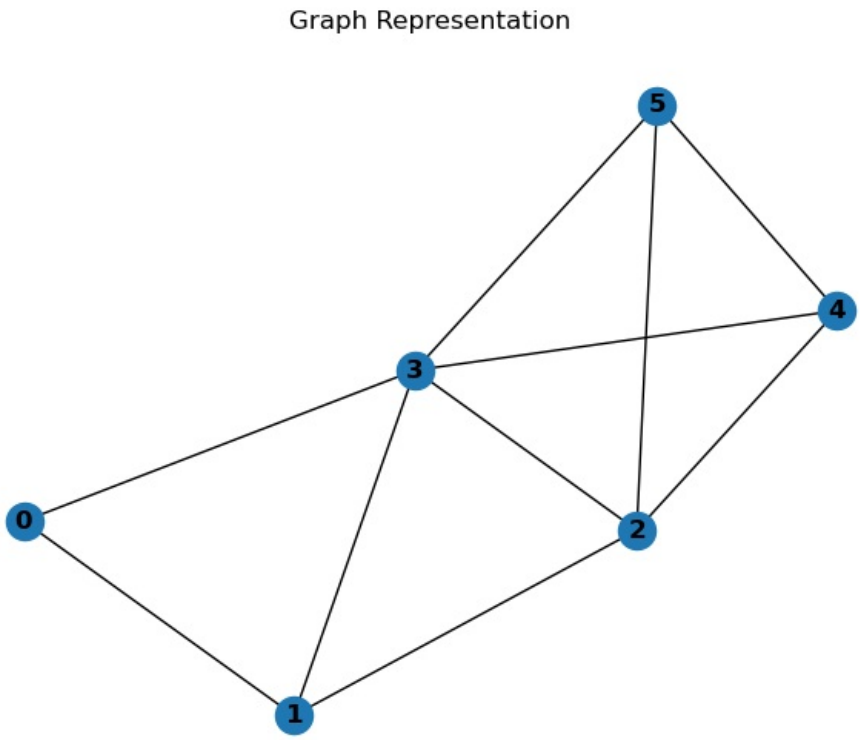
In [1]:

```
import networkx as nx
import matplotlib.pyplot as plt
from collections import deque

def dfs(graph, node, visited):
    if node not in visited:
        print(node, end=' ')
        visited.add(node)
        for neighbor in graph[node]:
            dfs(graph, neighbor, visited)

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

    while queue:
        current = queue.popleft()
        print(current, end=' ')

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

def main():
    vertices = int(input("Enter the number of vertices: "))
    edges = int(input("Enter the number of edges: "))

    graph = {}

    print("Enter edges (u v):")
    for _ in range(edges):
        u, v = map(int, input().split())
        graph.setdefault(u, []).append(v)
        graph.setdefault(v, []).append(u)

    # Displaying the graph
    G = nx.Graph()
    for u, neighbors in graph.items():
        G.add_edges_from((u, v) for v in neighbors)

    pos = nx.spring_layout(G)
    nx.draw(G, pos, with_labels=True, font_weight='bold')
    plt.title("Graph Representation")
    plt.show()

    start_node = int(input("Enter the starting node for traversal: "))

    # Depth First Search
    print("\nDepth First Search:")
    dfs(graph, start_node, set())

    # Breadth First Search
    print("\n\nBreadth First Search:")
    bfs(graph, start_node)

    plt.show()

if __name__ == "__main__":
    main()
```

```
Enter the number of vertices: 6
Enter the number of edges: 10
Enter edges (u v):
0 3
0 1
3 5
5 4
4 2
2 1
1 3
2 5
3 4
3 2
```




```
Enter the starting node for traversal: 1

Depth First Search:
1 0 3 5 4 2

Breadth First Search:
1 0 2 3 4 5
```

In [ ]: