In [3]:
```
class Graph:
    def __init__(self):
        self.graph = {}

    def add_edge(self, u, v):
        if u in self.graph:
            self.graph[u].append(v)
        else:
            self.graph[u] = [v]

    def dfs(self, start, visited=None):
        if visited is None:
            visited = set()
        visited.add(start)
        print(start, end=" ")

        for neighbor in self.graph.get(start, []):
            if neighbor not in visited:
                self.dfs(neighbor, visited)

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

        while queue:
            node = queue.pop(0)
            print(node, end=" ")

            for neighbor in self.graph.get(node, []):
                if neighbor not in visited:
                    queue.append(neighbor)
                    visited.add(neighbor)

def main():
    g = Graph()
    g.add_edge(0, 1)
    g.add_edge(0, 2)
    g.add_edge(1, 3)
    g.add_edge(1, 4)
    g.add_edge(2, 5)

    while True:
        choice = input("Enter '1' for DFS or '2' for BFS (or 'q' to quit):

        if choice == '1':
            print("DFS starting from vertex 0:")
            g.dfs(0)
        elif choice == '2':
            print("BFS starting from vertex 0:")
            g.bfs(0)
        elif choice.lower() == 'q':
            break
        else:
            print("Invalid choice. Please enter '1' for DFS, '2' for BFS, o

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

```
Enter '1' for DFS or '2' for BFS (or 'q' to quit): 1
DFS starting from vertex 0:
0 1 3 4 2 5 Enter '1' for DFS or '2' for BFS (or 'q' to quit): 2
BFS starting from vertex 0:
0 1 2 3 4 5 Enter '1' for DFS or '2' for BFS (or 'q' to quit): q
```

In [ ]: