python

```python
class Graph:
    def __init__(self):
        self.adjacencyList = {}

    def addVertex(self, v):
        if v not in self.adjacencyList:  # Check if vertex exists before adding
            self.adjacencyList[v] = []

    def addEdge(self, v1, v2):
        self.adjacencyList[v1].append(v2)  # Use append() instead of push()
        self.adjacencyList[v2].append(v1)  # Use append() instead of push()

    def bfs(self, v):  # Add self parameter to bfs() method
        queue = [v]
        visited = {}
        result = []

        visited[v] = True  # Use True instead of true

        while len(queue) > 0:
            currNode = queue.pop(0)
            result.append(currNode)

            for neighbour in self.adjacencyList[currNode]:
                if neighbour not in visited:  # Check if neighbour is not visited
                    visited[neighbour] = True
                    queue.append(neighbour)  # Add neighbour to queue

        return result

graph = Graph()

graph.addVertex("A")
graph.addVertex("B")
graph.addVertex("C")
graph.addVertex("D")
graph.addVertex("E")
graph.addVertex("F")

graph.addEdge("A", "B")
graph.addEdge("A", "C")
graph.addEdge("B", "D")
graph.addEdge("C", "E")
graph.addEdge("D", "E")
graph.addEdge("D", "F")
graph.addEdge("E", "F")

print("BFS result:", graph.bfs("A"))  # Use print() instead of console.log()
```