例 1.13 寻找欧拉回路算法实现

算法功能:寻找欧拉回路

修改参数：eulerEdges：图中边的连接关系信息
start:开始点
"""

```
def isEuler():
    allVisited = True
    for e in visited:
        if e == 0:
            allVisited = False
    if allVisited:
        if queue[0] == queue[len(queue) - 1]:
            return 1
        else:
            return 2
    return 0

def printPath(flag):
    if flag == 1:
        print("是欧拉回路:", end="")
    else:
        print("是欧拉道路:", end="")
    for i in range(len(queue)):
        if i < len(queue) - 1:
            print(queue[i], "-> ", end="")
        else:
            print(queue[i])



# 搜索过程只保存一条路的状态的信息，搜索结束后 queue,visited 会恢复为初始状态
def dfs(u):
    queue.append(u)
    flag = isEuler()  # 判断当前路径是不是欧拉路，如果是则打印
    if flag > 0:
        eulerFlag = True
        printPath(flag)
    for i in range(len(eulerEdges)):
        if visited[i] == 1:
            continue
        edge = eulerEdges[i]
```

```
        if edge[0] == u:
            visited[i] = 1
            dfs(edge[1])
            queue.pop()  # 将搜索过的点弹出队列
            visited[i] = 0  # 重置访问状态
        elif edge[1] == u:
            visited[i] = 1
            dfs(edge[0])
            queue.pop()  # 将搜索过的点弹出队列
            visited[i] = 0  # 重置访问状态

if __name__ == "__main__":
    eulerEdges = [
        (1, 3),
        (1, 5),
        (2, 2),
        (2, 3),
        (2, 5),
        (3, 4),
        (3, 5),
        (4, 5)
    ]
    start = 1  # 如果是欧拉道路必须从奇点开始
    visited = [0 for i in range(len(eulerEdges))]  # 访问过的路
    queue = []  # 保存路径信息
    eulerFlag = False
    dfs(start)
    if not eulerFlag:
        print("不是欧拉回路或欧拉道路")
```