例 1.8 Floyd 算法实现

算法功能:求解起始点和终止点的最短距离及路径

修改参数：node：节点列表
node_list:节点邻接距离关系（一组组输入）
from_code:起始节点
to_code:终止节点"""

```
class Floyd_Path():
    def __init__(self, node, node_map, path_map):
        self.node = node
        self.node_map = node_map
        self.node_length = len(node_map)
        self.path_map = path_map
        self._init_Floyd()

    def __call__(self, from_node, to_node):
        self.from_node = from_node
        self.to_node = to_node
        return self._format_path()

    def _init_Floyd(self):
        for k in range(self.node_length):
            for i in range(self.node_length):
                for j in range(self.node_length):
                    tmp = self.node_map[i][k] + self.node_map[k][j]
                    if self.node_map[i][j] > tmp:
                        self.node_map[i][j] = tmp
                        self.path_map[i][j] = self.path_map[i][k]

        print('_init_Floyd is end')

    def _format_path(self):
        node_list = []
        temp_node = self.from_node
        obj_node = self.to_node
        print(f"the shortest path is:{self.node_map[temp_node][obj_node]}")
        node_list.append(self.node[temp_node])
        while True:
            node_list.append(self.node[self.path_map[temp_node][obj_node]])
            temp_node = self.path_map[temp_node][obj_node]
            if temp_node == obj_node:
                break;
```

```
        return node_list


def set_node_map(node_map, node, node_list, path_map):
    for i in range(len(node)):
        # 对角线为0
        node_map[i][i] = 0
    for x, y, val in node_list:
        node_map[node.index(x)][node.index(y)]                                    =
node_map[node.index(y)][node.index(x)] = val
        path_map[node.index(x)][node.index(y)] = node.index(y)
        path_map[node.index(y)][node.index(x)] = node.index(x)


if __name__ == "__main__":
    INF_val=float('inf')
    node = ['1', '2', '3', '4', '5', '6']
    node_list = [('1', '2', 1), ('1', '3', 2), ('2', '3', 1), ('2', '4',
3),('2','6',7),
                 ('3', '4', 1), ('3', '5', 2), ('4', '6', 3), ('5', '6', 6)]

    # node_map[i][j] 存储i到j的最短距离
    node_map = [[INF_val for val in range(len(node))] for val in range(len(node))]
    # path_map[i][j]=j 表示i到j的最短路径是经过顶点j
    path_map = [[0 for val in range(len(node))] for val in range(len(node))]

    set_node_map(node_map, node, node_list, path_map)

    # 选取开始节点和终止节点，如果需要遍历则设置循环 for index,code in
enumerate(from_node)
    from_node = node.index('1')
    to_node = node.index('6')
    Floydpath = Floyd_Path(node, node_map, path_map)
    path = Floydpath(from_node, to_node)
    print(path)
```