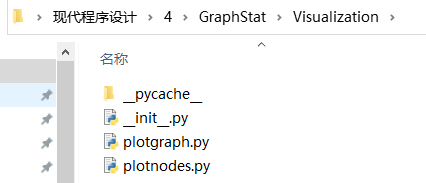
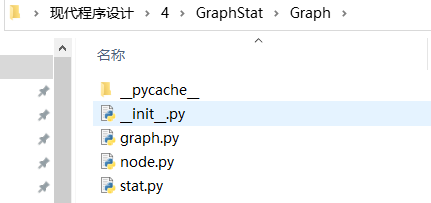
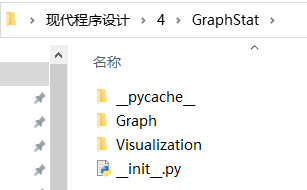
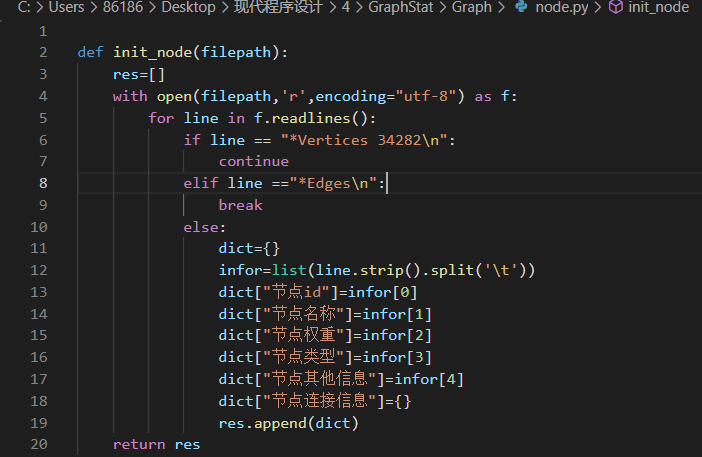
1. 首先建立包，分为图像信息处理和可视化两部分，如下图

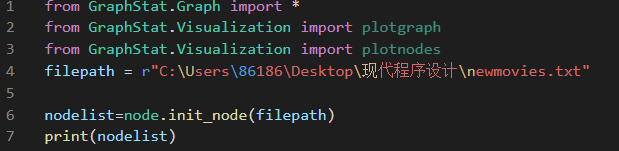


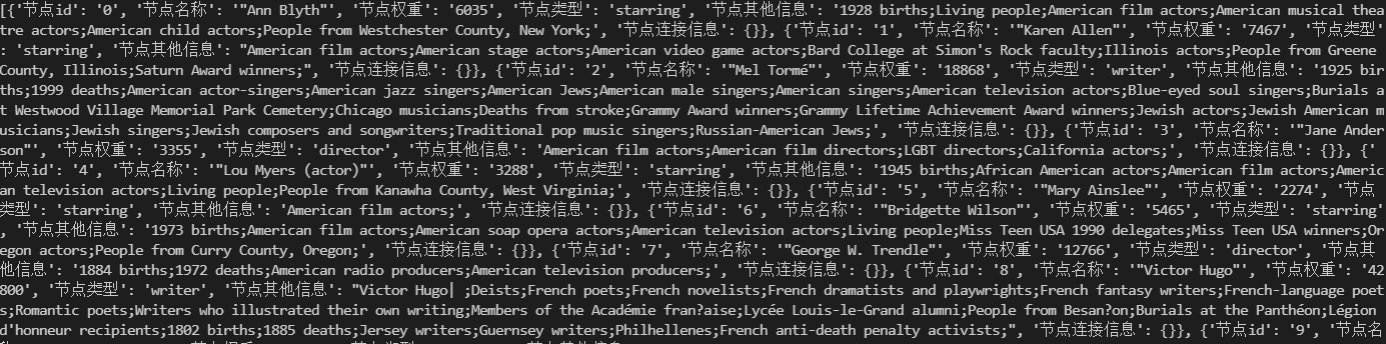
（1）节点信息处理

利用字典，将节点的属性储存在字典中，每个字典是一个节点，将所有的节点以列表的形式储存，实现函数：init\_node（）



调用部分结果：

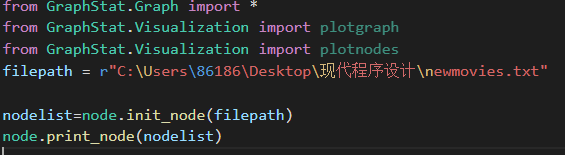


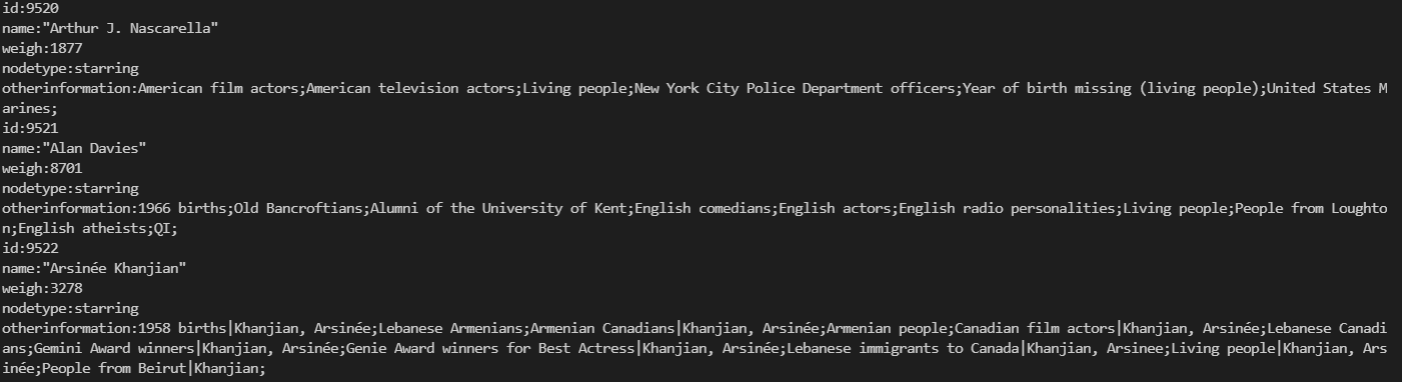


（2）节点信息输出

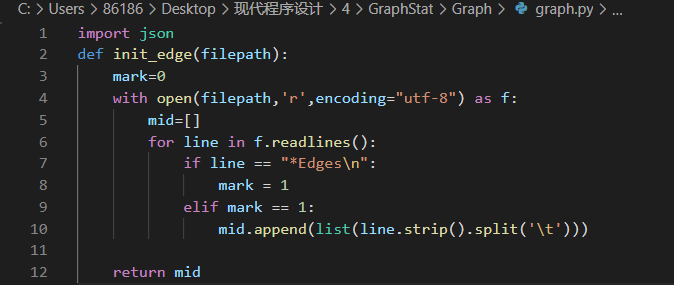
利用字典和format函数，将节点信息输出，实现函数：print\_node（）

调用部分结果如下：





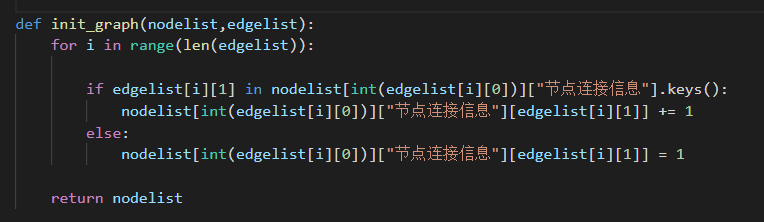
1. 图的信息处理
2. 初始化边信息，将边的信息以列表形式储存



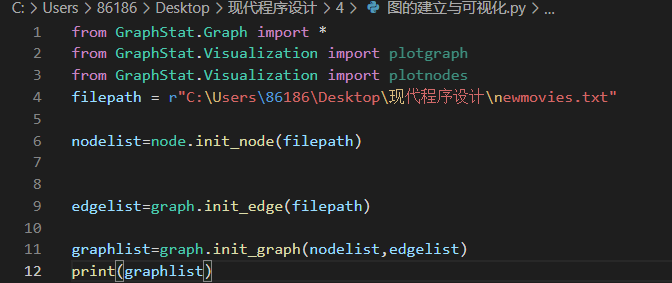
运行结果：

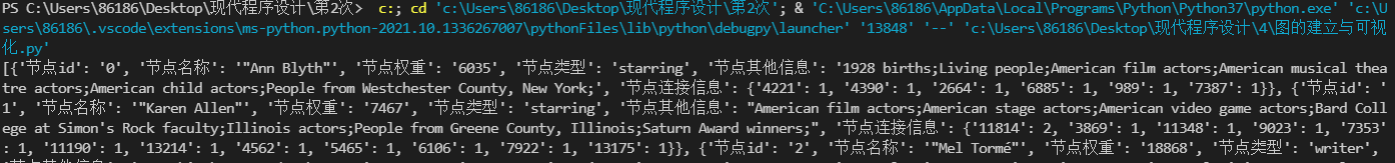


1. 去掉重复边，计算权重，同时将节点信息和边信息整合在字典中，每个节点增加一个“邻居节点”的属性，以字典方式储存，键为与该节点相邻的节点，值为这两点形成的边的权重。



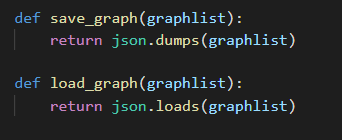
运行结果：





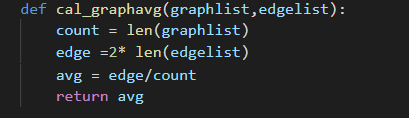
1. 图信息的序列化储存和反序列化加载

为了方便图信息的储存，设计序列化函数

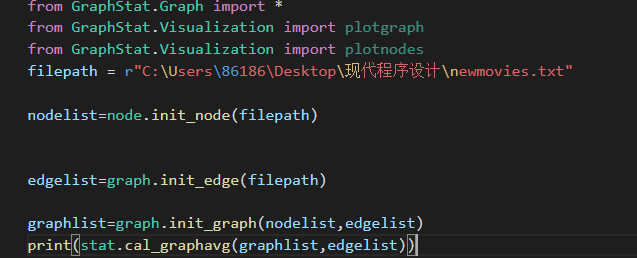


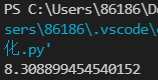
1. 图的信息统计
2. 计算图的平均度

平均度为所有节点度之和除以节点个数，而所有节点度之和等于边的数量的2倍，由此可以计算平均度

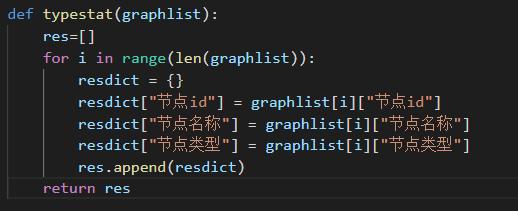


运行结果：



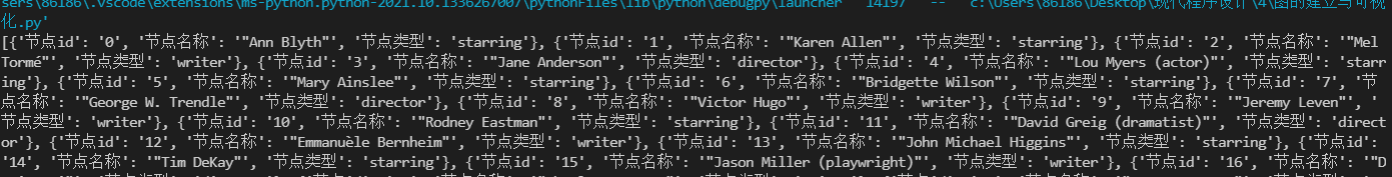


1. 获取节点的类型分布，单个节点以字典形式储存，所有节点以列表形式输出

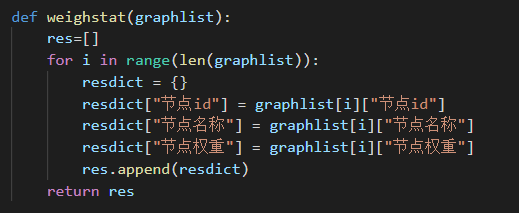


运行结果：

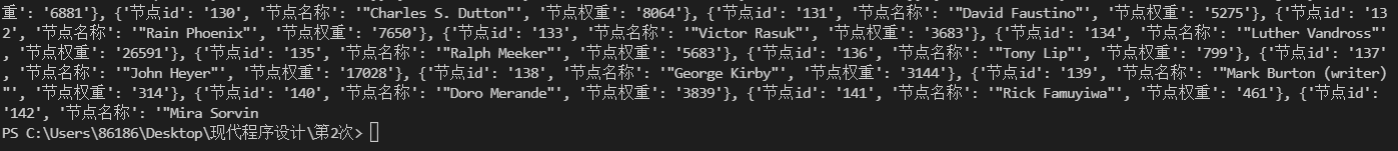




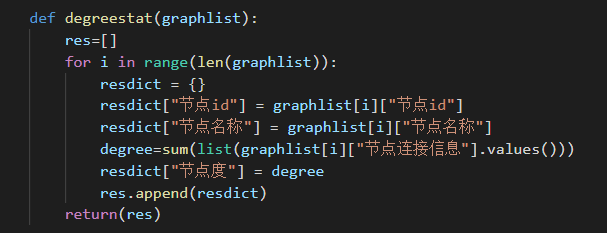
1. 获取节点的权重分布



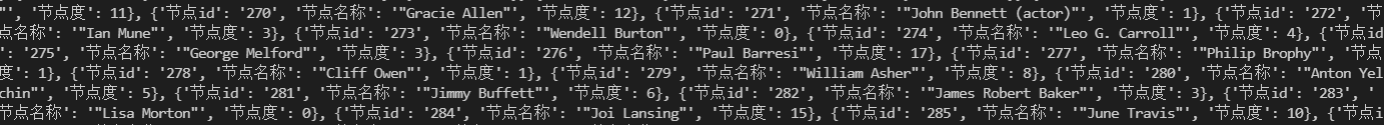
运行结果：



1. 获取节点的自由度分布

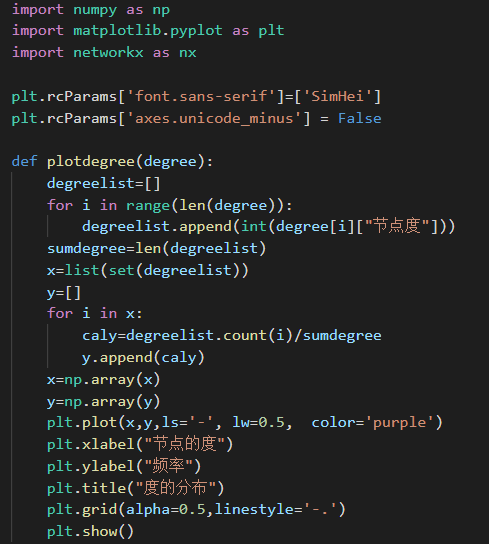


运行结果：

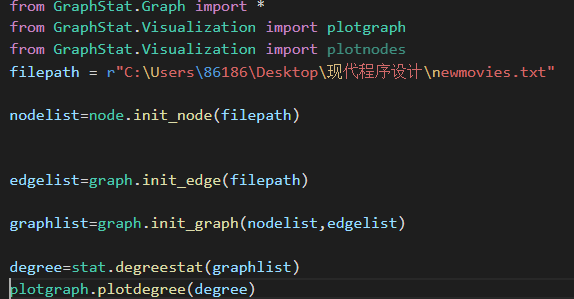


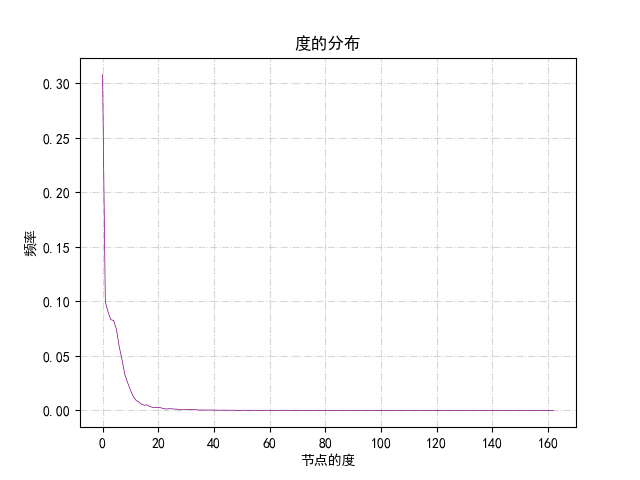
1. 图的可视化包
2. 图的自由度分布可视化

统计每个节点自由度的数量，计算出现的频率，以折线图的形式呈现图的自由度分布



运行结果：



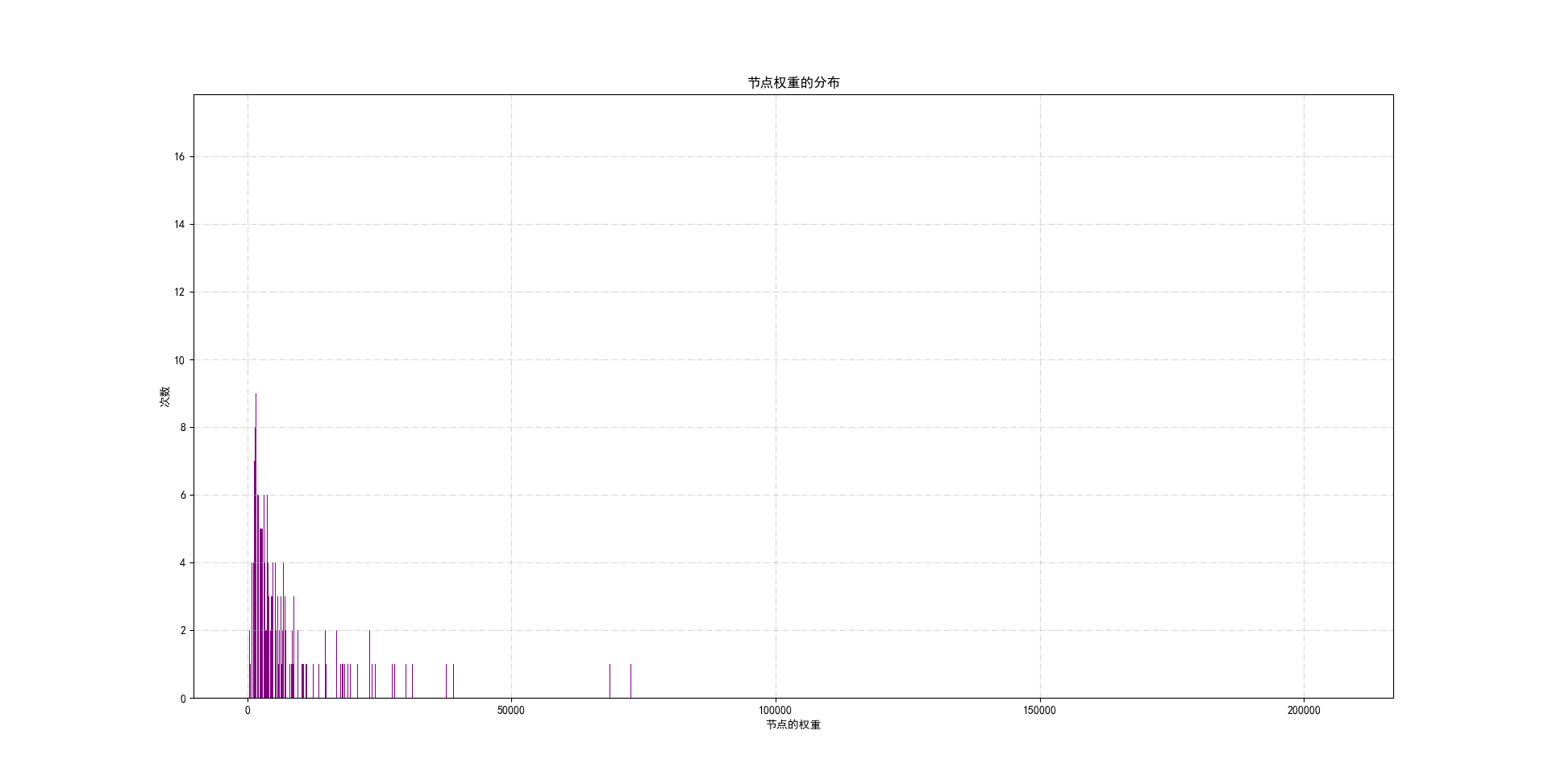


1. 节点的权重分布可视化

统计不同权重出现的次数，以柱状图的形式呈现结果



运行结果：

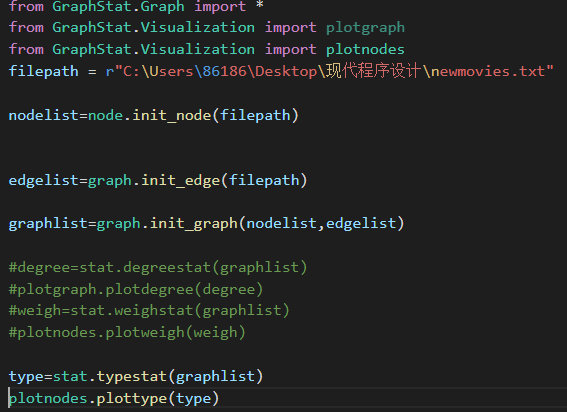


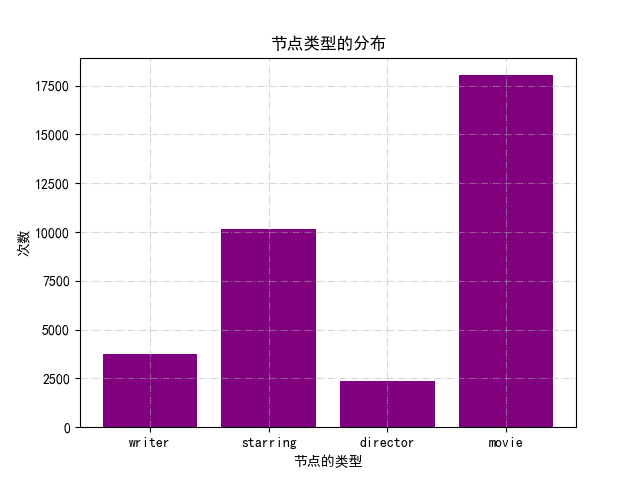
1. 节点类型分布可视化

统计不同类型节点的数目，以柱状图形式呈现



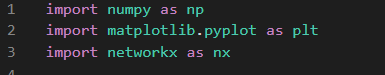
运行结果：

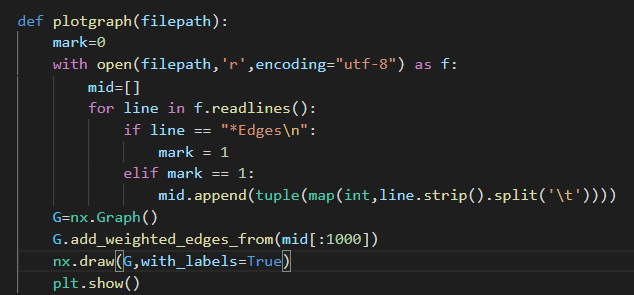




1. 利用networkx库，画出部分节点之间的网络结构

此处取前1000个节点，按照有权无向图画出关系，并且标出节点id





运行结果：

