Some general things to think about/talk about:

- What does the method do? Explain what the role of the different parameters is, what is returned and how the method operates.
- Show some examples both connected and unconnected.
- What is the running time of the method with respect to $|V|$ the number of vertices and $|E|$ the number of edges?
  * How many times is each vertex visited?
  * How many times is each edge visited/examined?

Some specific things to think about/talk about:

- Why are the visited and adjMap passed by reference? Why can we pass the adjMap parameter as a const, but not the visited parameter?
- What do the if statements in both methods do?
- What does adjMap.find(v)->second do?
- What is the for loop checking in grop.isConnected?
void dfs(int v, set<int>& visited, const map<int, list<int> >& adjMap){
    visited.insert(v);
    list<int> nbrList = adjMap.find(v)->second;

    list<int>::iterator nbr;

    for (nbr = nbrList.begin(); nbr != nbrList.end(); nbr++){
        if (visited.count(*nbr) == 0){
            dfs(*nbr, visited, adjMap);
        }
    }
}

bool grop_isConnected(const map<int, list<int> >& adjMap){
    set<int> visited;
    dfs(adjMap.begin()->first, visited, adjMap);

    bool connected = true;

    for(map<int, list<int> >::const_iterator it = adjMap.begin();
        it != adjMap.end();
        it++ ){
        if( visited.count(it->first) == 0 ){
            connected = false;
        }
    }

    return connected;
}