YIYANG ZHAO CS#@# HW4

Report:

Complete the code:

It only takes a few lines to have the code completed.

1) create a graph:

since we don’t really have a weight, we set it to 1 on each edge

```
for (int i=0; i<titles.size();i++){
        for (int j=0; j<links[i].length;j++){
                g.setDirectedEdge(i, links[i][j], 1);
        }
}
```

2) run Choi’s Dijkstra algothrism to find the shortest path.

```
Integer[] path=  d.getShortestPath(g, source, target);
```

The return of this function is a array (linked list) and

3) we traverse it to get the answer

```
while (true){
        counter++;
        System.out.println(thisint+" : "+titles.get(thisint));
        if (path[thisint]!= null) {

                thisint = path[thisint];
        } else break;

}
```

It is working fine… Some sample results

```
Finding path from 0 : For_Your_Consideration to 0 :
For_Your_Consideration
0 : For_Your_Consideration
length=0
No path

Finding path from 0 : For_Your_Consideration to 10 : Foster_care
0 : For_Your_Consideration
3741 : Television_network
4262 : United_States
1583 : Madonna_(entertainer)
10 : Foster_care
length=4

Finding path from 15 : Fox_Plaza_(Los_Angeles) to 20 :
Fracture_(2007_film)
```

```
15 : Fox_Plaza_(Los_Angeles)
1500 : Los_Angeles,_California
3800 : The_CW_Television_Network
2147 : New_Line_Cinema
20 : Fracture_(2007_film)
length=4

Finding path from 45 : Frau_Farbissina to 999 : July_1999
45 : Frau_Farbissina
4512 : Wikiquote
1009 : July_2007
999 : July_1999
length=3

Finding path from 666 : Iraqi_insurgency to 888 : Johanna_Sällström
666 : Iraqi_insurgency
length=0
No path

Finding path from 789 : January_26 to 1230 : Ladder_49
789 : January_26
777 : January_2002
580 : Impostor
935 : John_Travolta
1230 : Ladder_49
length=4

Finding path from 78 : Futsal to 12 : Four_Weddings_and_a_Funeral
78 : Futsal
2111 : National_sport
1011 : July_21
3040 : Robin_Williams
12 : Four_Weddings_and_a_Funeral
length=4

Finding path from 798 : January_6 to 152 : Get_Down_Tonight
798 : January_6
788 : January_25
1088 : KC_and_the_Sunshine_Band
152 : Get_Down_Tonight
length=3
Finding path from 321 : Hallgeir_Brenden to 905 : John_E._McLaughlin
321 : Hallgeir_Brenden
3297 : September_21
2334 : October_2008
2271 : November_2004
```

```
905 : John_E._McLaughlin
length=4

Finding path from 1000 : July_2 to 1001 : July_20
1000 : July_2
1001 : July_20
length=1

Finding path from 1001 : July_20 to 1002 : July_2000
1001 : July_20
1002 : July_2000
length=1
```

Observations:

1) When target=source, the length of the path is always 0. This makes sense because we don’t want loops and the shortest path from one node to itself is not going anywhere.
2) A—> B can be very different from B—> A in terms of both path and degree. This is because
    1) This is a directed graph so path might not be reversible
    2) Even it is, there might be other route from B—>A that is shorter than the reverse of A—B
    3) And the algorithm we find path is finding a shortest one, there might be many
3) A few pairs are impossible to reach
4) Other than those mentioned in 3), most pairs are reachable in less than 4 steps.
    1) I randomly chose 1000 pairs and the results are:
        1) Most of them (~700) case has degree 4
        2) A few (~150) has degree 3
        3) A few (~100) is not possible
        4) Fewer are of degree 2 or 1 or 5
        5) NO case has degree >=6. Maybe there can be some but in my experiment I didn’t find any of them. That should be very few if at all exits. So it might be degree of 5
5) Although the graph is very sparse, it is still possible to have most pairs connected.
6) Finding one path is a very fast process and is instant.