CS 114, Fall 2017, Prof. Calvin
Problem set #11
Part 1 due: noon, Tuesday Nov. 21.
Part 2 due: in class, Tuesday Dec. 5.

For this last assignment you can work alone or in pairs. If you work with someone else put both names in comments on the top of any files that you submit. Each of the two parts counts the same as a regular assignment. In addition, the solution to part 2 with the best running time will receive a bonus equal to the maximum score on a regular assignment, *if* the author(s) can clearly explain their solution to the class.

For many years people have been interested in the question of how connected social networks are. A famous experiment was carried out by Stanley Milgram in 1967. The experiment went something like this: Randomly chosen people in Omaha were given packets that included a letter addressed to a randomly chosen person in Boston. In the unlikely event that the resident of Omaha new the recipient, he was to send it directly. Otherwise, he was asked to select someone he knew who was more likely to know the recipient and send it on to them. The question: On average, how many intermediaries before the letter reached its final destination? The answer turned out to be about 6. See `http://en.wikipedia.org/wiki/Small-world_experiment`.

This kind of experiment can be carried out nowadays on computer social networks that have a relation between members ("friends" on facebook, "contacts" on LinkedIn, etc.). If you take two members at random, how close are they? For example, if you take two random members of facebook, what is the length of the shortest chain of friend links connecting them?

Social networks are unlikely to share this kind of information, but we can perform the same type of experiment using the IMDB actor data. Let's say that if two actors worked on the same film, they are distance 1 apart. If actor 1 worked on a film with actor 2, and actor 2 worked on a film with actor 3, who worked on a film with actor 4, then we would say that the distance between actor 1 and actor 4 is 3 if there is no shorter path connecting them.

In this exercise we will work with the files actresses.list and actors.list, which contains, for each actor in the database, a list of movies the actor appeared in. Here is a typical entry:

```
Cronin, Doug (I)        Chasing Dreams (1982)  [Sheriff]  <28>
                        Scared Straight! (1978)
```

The movie information comprises several items besides movie name (date, role, tv or film, order in credits, etc).

How would you store the data from actors.list in a Java program? You might start by writing a class to represent the data for a particular actor; lets call the class ActorRecord. This class should store the actor's name, and then a list of the movies that the actor has appeared in. The length of the movie list will vary widely, but if we store it in an `ArrayList<String>` this will not present a problem. We can store the actor records in another ArrayList, this time it will be an `ArrayList<ActorRecord>`.

The actresses and actors files are quite large, and so we will work with compressed versions,actresses.list.gz and actors.list.gz, which are on Moodle. Also in

that subdirectory is a class RetrieveActors. The constructor takes a string argument that specifies the file from which to extract the actor data; you should give it the path name to actresses.list.gz and/or actors.list.gz. This class reads the compressed actor data without decompressing the entire file. It contains a method getNext() that returns a string for the next actor, or null if there are no more actor records in the file. The string comprises a list of strings separated by "@@@". The first string is the actor's name, and subsequent strings are the movies that the actor appeared in. Each movie string has two initial characters; a "TV" indicates that the title is a made-for-TV movie, a "TS" indicates a TV series or mini-series, a "VO" indicates a video, and "FM" indicates a film.

For example, the entry displayed above would be returned as

```
Cronin, Doug (I)@@@FMChasing Dreams (1982)@@@FMScared Straight! (1978)@@@
```

You do not need to be familiar with the details of how RetrieveActors works, or of the format of the IMDB files.

The program BusyActress.java reads in the data from actresses.list.gz and determines which actress appeared in the most movies.

Write a program that creates a graph using the data in actresses.list.gz and actors.list.gz. You should have a vertex for each actor and an edge between two actors if they worked on a film together. (Consider only films; not tv shows or videos.) Choose your favorite actor, and compute the "distance" to each other actor. For each possible distance, give the number of actors that distance away (take the distance to be infinity if there is no path.) Use the graph classes on Moodle. That directory also contains a file shortestActors.list.gz, which is of the same format as the other actor files but with only 8 actors and 5 movies. You can check the distances you obtain with the solution in ActorsSmallWorld.pdf.

Call your program SmallWorld. For part 1 your program should work for shortestActors.list.gz. For part 2, tune your program so that it works with the combined data from actresses.list.gz and actors.list.gz. You may need to reevaluate your choices of data structures and algorithms to deal with the large data set. Use only algorithms and data structures (and your modifications) that we have covered in the course.

Here is the table for Hedy Lamarr based on all actresses and actors:

dist.	count
1	2,224
2	196,781
3	1,555,043
4	974,718
5	121,062
6	12,232
7	1,358
8	321
9	46
10	7
∞	1,287,288

Notes:

(1) In a January 1994 Premiere magazine interview about the film "The River Wild", Kevin Bacon commented that he had worked with everybody in Hollywood or someone who's worked with them. This claim attracted a lot of attention, even resulting in a parlor game called "Six Degrees of Kevin Bacon"; for a discussion, see `http://www.cnn.com/2014/03/08/tech/web/kevin-bacon-six-degrees-sxsw/`.

(2) If you type in "bacon number *name*" in google, it will give you the "bacon number" of the name you enter, and also give you the path that results in that number. You can try checking your program with the results from Google, but you might get some different answers. If you are interested in the reason for that, see `http://nymag.com/daily/intelligencer/2012/09/google-bacon-number-kevin-bacon.html`.

(3) In addition to appearing in many films, Hedy Lamarr (born Hedwig Eva Maria Kiesler in 1914, died 2000) patented (with George Antheil) a radio frequency hopping communications system that introduced some of the principles upon which modern wireless communications systems are based. See `http://invent.org/inductees/markey-hedy/`.

Exercises: (Not to turn in.)

(1) Use induction to prove that a (simple) graph $G = (V, E)$ has at most $|V|(|V|-1)/2$ edges.

(2) Suppose that you need to be able to quickly determine if two given vertices of a graph are connected by an edge. Which graph representation would you use, adjacency list or adjacency matrix? What are the time and space requirements?

(3) Consider a communications network that is a free tree; that is, a graph with no cycles. The time delay in a signal sent across a path is a small constant times the number of links in the path. Suppose that you want to determine the maximum possible delay in the network. The *diameter* of a tree T is the length of the longest path between two vertices of T. Describe an efficient algorithm for computing the diameter of T.