Homework #3: Graphs, Max Bridge
Due Date: written by 5pm Friday 3/2, program by 5pm 3/3

Written Part

Try to solve these problems, and write out solutions on paper.

Problem 1. Find the paper "More Robust Hashing: Cuckoo Hashing with a Stash." Summarize their main result in a paragraph. In particular, what goes in the "stash"? Give (at least) one argument in favor of using this idea, and (at least) one against.

Problem 2. Using hashing and array doubling, describe a graph representation with these features:

- There are methods telling us V and E, the current number of edges and vertices. The vertices are integers, 0 to $V-1$. The data structure uses $O(V+E)$ space (words of memory).
- When we construct a graph, it is empty (no vertices, no edges). A method that lets us *create* a new vertex with no edges, in O(1) expected time. (If there were V vertices before, the new vertex is number V.)
- Given a vertex v, we can list its neighbors in O(1) time per neighbor.
- A pair of vertices (u, v) identifies a potential edge. We can *find*, or *add*, or *remove* such an edge in $O(1)$ expected time.

Here "expected" includes both randomization (averaging over random choices, like in hashing) and amortization (averaging over the sequence of operations). We'll assume a good hash function.

Program Part: Max Bridge

In an undirected graph G, a *bridge* is an edge $e = \{u, v\}$, such that the deletion of e disconnects u and v (there is no path from u to v in $G - e$). Let s_u be the size of the component of u in $G - e$, and similarly let s_v be the size of the component of v in $G - e$. Say the *size* of the bridge e is $\min(s_u, s_v)$. We are interested in finding a bridge e in G of maximum size.

You are to write a program `MaxBridge.java`. Running `java MaxBridge` should read the file `movies.txt` (in the current directory), and print out a description of the bridge e of maximum size. See hw3/Movies.java for an example program reading `movies.txt`.

Your program should print out three lines of text: the name of the movie (one endpoint of bridge e), the name of the actor (the other endpoint of e) and the size of e (an integer). The first two lines (the movie and actor) may be reversed. For example this is incorrect output, but in the correct format:

```
North by Northwest (1959)
Grant, Cary
17
```

(The correct answer will not be nearly so famous.)

Besides the file `movies.txt` in the current directory, your program may also assume that share/hw3/book.jar is on the Java CLASSPATH, so you may use any of the classes defined there (it contains compiled versions of everything in share/book/). Or if you prefer, you may just modify any source code you like from share/book/, and incorporate that into your program.

Do not hard-code your answer, since I will also test your program on modified versions of `movies.txt`.

When you think you have found the correct output for the default `movies.txt` file, email it to me (mic@mathcs.emory.edu). I'll offer a small point bonus (and recognition, if you like) to the fastest correct solver.

For more details on how to copy the files and turnin, see hw3/Notes.txt. I'll also probably distribute some advice by email to the class mailing list.

Honor Policy Your work for this class is governed by the **Emory Honor Code**[1]. Programming work is also governed by the **Math/CS SPCA**[2] (Statement of Policy on Computer Assignments). In particular, you should take care to protect the confidentiality of your homework files. Apparent honor code violations will be referred to the Emory Honor Council. If you have questions about what is allowed under the policy, please ask.

[1] http://college.emory.edu/current/standards/honor_code.html
[2] http://mathcs.emory.edu/spca.php