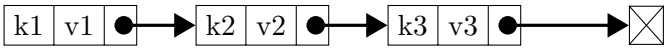
CS 240 Homework 8

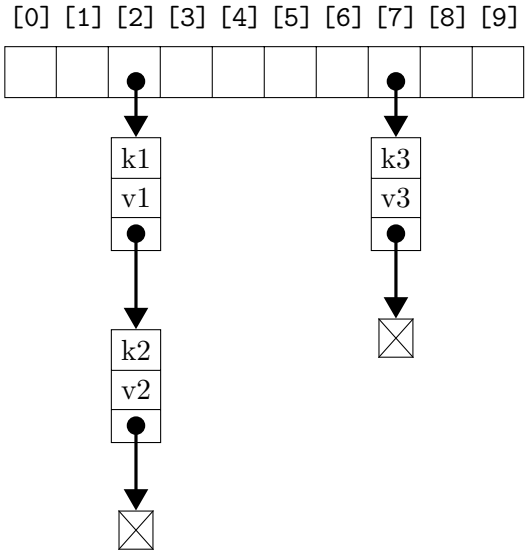
Alex Vondrak

Due: March 9, 2012

This homework will have you modify Homework 7 to use a more efficient version of the `AssociationList<K, V>`. Namely, you'll convert it into a hash table using *chained hashing*. Instead of storing a "flat" linked list of your key/value pairs that you have to search through each time you want to `get` something



you keep an array of linked lists:



To insert a key/value pair into the `AssociationList`, you must use the `hashCode` method of the key (it's built into every Java object, so should be suitable for our use), and simply mod the **`int`** it returns by the capacity of your array. This gives you the index in the array of the linked list to which we add the key/value pair's node. If we're lucky, the keys will be well-distributed across the array. In the best case, we'll just have an array of single-element linked lists. That way, the `get` method need only look up which index a key should reside at, and do little-to-no searching of the linked list at that index.

In summary, modify the fields, constructor, `get` method, and `set` method of the `AssociationList<K, V>` class to turn it into a hash table.

Input Format

Input is of the same format as Homework 7.

Output Format

Output is of the same format as Homework 7.

Input Sample

Test against the ACID input available at `http://www.csupomona.edu/~ajvondrak/cs/240/12/winter/hw/hw8.in`

Output Sample

Test against the ACID output available at `http://www.csupomona.edu/~ajvondrak/cs/240/12/winter/hw/hw8.out`

Extra Credit

For extra credit, upload a plain-text file `hw8.ec` to your ZFS share along with `hw8.java`. Strictly, `hw8.ec` should be a Java file, but I'll be using the different extensions to distinguish it from your "main" Homework 8.

In `hw8.ec`, you should include all the specified code **and comments** from the Project 3 I gave in my Fall 2011 CS 240 class: `http://www.csupomona.edu/~ajvondrak/cs/240/11/fall/prj3.pdf`

Note that you don't need to implement a hash table for this; you should just use Java's `java.util.HashMap` object, whose documentation is available at `http://docs.oracle.com/javase/6/docs/api/java/util/HashMap.html`

Also, this code won't be compiled/run against any ACID output, so you needn't follow a particular format. (Your running times are almost guaranteed to be different from mine anyway.) Just implement your code, and explain your output & results in the comments.