CSCI 321 Computer Science III Summer 2019

Assignment 4

Problem: ***Build a hash table for websites***

General Setup and Requirements:

* A website w consists of two fields: (url, name)
* Use w’s url to compute w’s hash key:
  + Hash code: url -> integer
  + Compression function: integer -> [0, m-1] where m is the size of the table
* In your hash table implementation, it should have at least three methods:
  + put(key k, value v): Put a new website by its key k to the hash table
  + get(key k): Return the website associated with k
  + delete(key k): Remove the website associated with k
* As a test suite, the following operations should be performed:
  + Put ([www.uscupstate.edu](http://www.uscupstate.edu), USC Upstate)
  + Put ([www.google.com](http://www.google.com), Google)
  + Put (www.yahoo.com, Yahoo)
  + Get [www.google.com](http://www.google.com)
  + Delete [www.yahoo.com](http://www.yahoo.com)
  + Get [www.uscupstate.edu](http://www.uscupstate.edu)
  + Get [www.google.com](http://www.google.com)
  + Get www.yahoo.com

**Part a**. Implement a hash table to store websites using **linear probing** to handle collision. Sample code “LinearProbingHashTable.java” is attached for your reference. It implements the hash table using linear probing. Attach your code and screenshots.

**Part b**. Implement a hash table to store websites using **double hashing** to handle collision. You can modify the Sample code in Part a for this part. Attach your code and screenshots.

**Note: You can refer to the class slides and the following to understand the idea of linear probing and double hashing.**

**Linear probing:** This technique is used when we have more index in the table then the values to be stored. Linear probing technique work on the concept of keep incrementing until you find the empty slot. The pseudo code looks like this.

index = h(k)

while( val(index) is occupied)

index = (index+1) mod m

**Double hashing technique:** In this technique, we use two hashing functions h1(k) and h2(k). If the slot at h1(k) is occupied then the second hashing function h2(k) used to increment the index. The pseudo-code looks like this.

index = h1(k)

while( val(index) is occupied)

index = (index + h2(k)) mod m