QUEENS COLLEGE Department of Computer Science
CSCI 313 Practice problems on Hashing and Hash Tables

Instructor: Alex Ryba

These problems were given on exams for this course. Some older problems did not make use of generics in Java, but generic implementations are now required in this course.

Problem 1 (a) Assume that a chained hash table is implemented with the following instance variables.

```
public class ChainedHashTable<K, V> implements Map<K, V> {
   int size;
   int capacity;
   ArrayList<Pair<K, V>> bucket[];
   // method implementation code omitted
}
```

Write a Java implementation for the method *get*

Answer:

(b) Consider these three possibilities for a hash function to be applied to String data (representing words in text). The data is to be stored in a chained table with capacity 20. For each function either describe any reason(s) why it would not work well or state that it is a good choice.

(i) The function converts the first character in the String to lower case and returns its ASCII code modulo 20.

Answer:

(ii) The function generates a random number x that indexes a position in the String. It then converts character number x in the String to lower case and returns its ASCII code modulo 20.

Answer:

(iii) The function converts all characters in the String to lower case and returns the sum of their ASCII codes.

Answer:

Problem 2 The generic class ChainedHashTable has the following partial implementation.

```
public class ChainedHashTable<K, V> implements Map<K, V> {
  int size;
  int capacity;
  ArrayList<Pair<K, V>> bucket[];

  ChainedHashTable(int cap) {
    capacity = cap;
    bucket = (ArrayList<Pair<K, V>>[]) new ArrayList[cap];
    size = 0;
    for (int i = 0; i < cap; i++)
      bucket[i] = null;
  }

  // standard methods omitted

}
```

Give a complete implementation of a standard method called *put*. Assume that the class K has a useful method called *hash* that returns a positive integer result.

Give a complete implementation of a non-standard method called *loadFactor* that returns the average number of elements in each bucket.

Problem 3 (a) Describe the instance variables that are used to store and implement a hash table that uses open addressing and linear probing. (Give Java declarations for the instance variables.)

(b) Using the instance variables that you gave in (a), implement the methods `public void insert(K k, V x)` and `V find(K k)`.