1. Define the following terms in your own words:

* Hashtable: hashtable is an efficient data structure that implements an associative array of abstract data types and can map keys to values and only occupies linear time. It uses a hash function to compute an index into an array of slots to insert the elements.
* Hash function: hash function is used to map the keys to the positions in the hashtables.
* Collision: Collision happens when a hash function maps two keys to the same address, and it is unavoidable.

2. Research and provide the answer to the following question (please cite your

resources)

* What is the difference between a Hashtable and a Hashmap?  
  Hashmap allows one null key and null values while hashtables do not. HashMap is non-synchronized and not thread-safe, while hashtable is synchronized, and thread-safe, therefore HashMap is fast and Hashtable is slow. HashMap is traversed by fail-fast Iterator and Hashtable is traversed by Enumerator and Iterator. Also, Hash map does not guarantee the order of the map will stay constant.
* What is the desired running time for search, insert and delete in a Hashtable?
  + The desire running time for search insert and delete is all in linear time, so O(1).

3. Provide two different ways to deal with collision and briefly explain both of them.

There are two ways to deal with collision: chaining and open addressing/linear probing.

**Chaining** stores all elements that hash to the same slot into a linked list, so all the values with the same key are in a long LinkedList. We store the point of the head of the linked list in the matching slot, therefore all the elements are still put in the matching slot.

**Open Addressing or linear probing** uses a systematic procedure to store the collided value into the next free slot, so all the elements are stored in the table without needing extra space. However, it does require more time when it comes to searching.

4. Similar to the fruits examples discussed in class, provide your own example that

illustrates storing strings using a hash function into a hashtable with linear probing.

(Your example should include collision scenarios)

Hash function: length % 6

List: animals

| Key | Length | Hash Value | Hash Table |
| --- | --- | --- | --- |
| Rabbit | 6 | 0 | 0 Rabbit → Turtle also maps to 0, so search for the next available slot |
| Lion | 4 | 4 | 1 Giraffe |
| Tiger | 5 | 5 | 2 Kangaroo |
| Kangaroo | 8 | 2 | 3 Turtle → next available slot found |
| Giraffe | 7 | 1 | 4 Lion |
| Turtle | 6 | 0 | 5 Tiger |

Here the collision happens as the hash value of turtle is the same as rabbit, so in linear probling, we map turtle to the next available slot, which is hash value = 3, and put it there.

Reference

Lecture Notes

<https://www.geeksforgeeks.org/differences-between-hashmap-and-hashtable-in-java/>

<https://www.javatpoint.com/difference-between-hashmap-and-hashtable>