1. Hashtable: A hash table is a data structure that uses array to map keys (such as numbers, strings) to hash values (usually indices). Hash tables offer very fast insertion and searching.

Hashfunction: Hash function is a function that maps data to fixed-size values (slots of the hash table).

Collision: Collision happens when a hash function maps two (or more) different keys to the same slot on the hash table.

1. What is the difference between a Hashtable and a Hashmap?

A Hashmap is not thread-synchronized while a Hashtable is. That means a Hashtable is thread-safe and can be shared with a number of threads. Also, Hashmap allows one null key and multiple null values while Hashtable does not allow any null keys or values.

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

What is the desired running time for search, insert and delete in a  
Hashtable?

The desired running time for those three operations in a Hashtable is O(1). Only under the circumstance where collision does not occur or where the key is not found.

Source: class slides & https://stackoverflow.com/questions/25194551/why-is-hashtable-delete-o1

1. The two ways of dealing with collisions is open addressing and chaining. Open addressing uses a systematic procedure to store keys in free slots that are left in the hash table when a collision occurs. Chaining stores multiple keys that have the same hash value in a linked list and has a pointer at the head of the linked list to solve collisions.
2. Keys: Lion, rabbit, tiger, dolphin, butterfly, dog, whale

value = hash(string) = number of Chars in string % 5

Chars(lion) % 5 = 4

Chars(rabbit) % 5 = 1

Chars(tiger) % 5 = 0

Chars(dolphin) % 5 = 2

Chars(butterfly) % 5 = 4

Chars(dog) % 5 = 3

Chars(whale) % 5 = 0

Inserting “lion”

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 |
|  |  |  |  | lion |  |  |  |

Inserting “rabbit”

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 |
|  | rabbit |  |  | lion |  |  |  |

Inserting “tiger”

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 |
| tiger | rabbit |  |  | lion |  |  |  |

Inserting “dolphin”

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 |
| tiger | rabbit | dolphin |  | lion |  |  |  |

Inserting “butterfly”: butterfly should be inserted at value 4; however, hash value 4 is already taken by “lion” so we move to the next free slot which is 5

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 |
| tiger | rabbit | dolphin |  | lion | butterfly |  |  |

Inserting “dog”

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 |
| tiger | rabbit | dolphin | dog | lion | butterfly |  |  |

Inserting “whale”: whale should be inserted at value 0; however, hash value 0 is already taken by “tiger” so we move forwards to find the next free slot. Value 1, 2, 3, 4, 5 are all taken by other strings, thus we can only insert “whale” at value 6.

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 |
| tiger | rabbit | dolphin | dog | lion | butterfly | whale |  |