Perfect Hashing

Hash Function maps each different key to distinct integers. Means hash table that uses a perfect hash has **no collisions**. This is the reason it is also known as **optimal hashing.**

In Perfect Hashing, we will get O(1) as the worst case time complexity in case of searching.

Every slot contains one hash table and for every slot 1 hash function is also there.

Load Factor : represents at what level the HashMap capacity should be doubled.

For example number of slots and load factor as 16 * 0.75 = 12. This represents that after storing the 12th key – value pair into the HashMap , its capacity becomes 32.

Given m number of slots and n number of keys load factor is **n/m (keys/slot).**

Open Addressing - 0 <= Load Factor <= 1

Chaining - 0 <= Load Factor <= infinity

The expected number of probes/attempts in an unsuccessful search of open addressing is :

1/(1-load factor)

The expected number of probes in successful search of open addressing is :

1/alpha ln 1/1 - alpha

Application of Hashing Data Structure - Bloom Filters

Bloom Filter is a probabilistic data structure that is totally based on Hashing. Used to add element to a set and check whether an element already present in the set or not. Element are not added to the set instead hash value of those elements are added.

Will either say that an element is definitely not in the set or that there is a high probability that the element is present in the set.

Competitive Programming - Need to compare lots and lots of strings best approach to be used.

1. Insert(k)

2. Lookup(k) - Yes or no

Probability of says "YES" when key k is not present{False Positives}

If we are using a bloom filter with **m bits** and **k hash function**, insertion and search will both take **O(k) time complexity**.

Space complexity - O(m)