Open Addressing

- ⇨ No. of slots in Hash Table ≥ No. of keys to be inserted
- ⇨ Cache Friendly

Linear probing

Linearly search for next empty slot when there is a collision.

0	49
1	50
2	51
3	16
4	56
5	15
6	19

50, 51, 49, 16, 56, 15, 19

hash(key) = key % 7

How to handle deletion in open addressing?

0	
1	50
2	51
3	15
4	
5	
6	

insert(50), insert(51), insert(15), search(15), delete(51), search(15)

hash(key) = key % 7

Search: We compute hash function, we go to the index and compare if we find, we return true. Otherwise we linearly search. We stop when one of the three cases arise, 1. Empty slot, 2. key found, 3. Traverse through the whole table.

Clustering (A problem with linear probing)

For 1 collision the clustering size would be 2

For K collision the clustering size would be k + 1

How to handle clustering problem with linear probing?

1. Quadratic Probing (Secondary Clusters)
   hash(key, i) = (h(key) + i^2) % m
2. Double Hashing
   hash(key, i) = (h_1(key) + i * h_2(key)) % m