Diagram, schematic

Description automatically generated

1. **Open Hashing** : The way to create a list and add it next.

**Pros :** Insertion and Deletion is O(1) (Insertion from the Head)

**Cons :** Searching and deletion will O(n), since it making a list. There will be a chances of unused slot and using extra spaces to make list.

**Load Factor alpha** = Elements or Keys / slots , as per above snap alpha = 5 / 5

**2.1. Linear Probing** : In this, it will add in next available location.

**(h + i)mod n,** here I is probe number means how many times you've checked the slot and n = 10

For Ex : Keys = 43,135,72,23,99,19,82 and hash function is (key mod 10),

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 19 |  | 72 | 43 # | 23 | 135 | 82 |  |  | 99 |
| 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 |

23 so (h+i)mod10, (3+1)mod10, 4 mod 10 = 4

19 so 9+1 mod 10, 10 mod 10 = 0

82 so 2+1 mod 10, 3 mod 10 = 3 (filled with 43), I = 2 -> 2+2 mod 10, 4 mod 10 = 4 (filled with 23), I = 3 -> 2+3 mod 10 = 5 (filled with 135), I=4 -> 2+4 mod 10=6

For Ex : 44 mod 6 = 2 and 2 index is already filled with 32, so it will find the sequential next available location means 3rd index. (based on above image)

**Pros :** No extra space required, since we're filling here the next slot

**Cons : 1.** search time : good case O(1) worst case O(n)  
 **2.** Deletion difficult, O(n) and one problem that usually happens while deletion, suppose in 1st Operation 43 has been deleted then 2nd operation need

to delete 82, so hash of 82 = 2 (match fail 72 present) then next \_ blank (43 has been deleted already) then search will be failed but still 82 is present

in hash. To avoid this issue, we add some special characters '#' to make the search in progress.

1. Primary clustering and Secondary clustering -> clustering means a group of element or a section of elements. So it takes time to cross one cluster.

**2.2. Quadratic Probing :** This follows the equation of

**(h + i^2 mod n)**.

Here, h is hash value, I is probe number means how many times you've checked the slot and n = 6

Ex based on above snapshot : Value -> 30 , 30 mod 6 = 0 but 0 index is already filled with 24, so follow the Quadratic probing.

(h + i^2 mod n)

(0 + 1^2 mod 6) = 1 i.e. already filled with 19

(0 + 2^2 mod 6) = 4 so value 30 will go in 4th index

Ex : Keys = 42,16,91,33,18,27,36,62

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 |
| 36 | 91 | 42 | 33 |  |  | 16 | 27 | 18 |  |

36 so hash of 36 is 6 (filled with 16), I = 1 so (6+1^2 mod 10) = 7 (filled with 27), i=2 so (6+2^2 mod 10)=0

62 so hash of 62 is 2 (filled with 42), I = 1 so (2+1^2 mod 10)=3 (filled with 33), i=2 so (2+2^2 mod 10)=6(filled with 16), i=3 so (2+3^2 mod 10)=1(filled with 91),

i=4 so (2+4^2 mod 10)=8 (filled with 18), i=5 so (2+5^2 mod 10)=7 (filled with 27), i=6 so (2+6^2 mod 10)=8 (filled with 18), i=7 (2+7^2 mod 10)=1 (filled with 91), i=8 (2+8^2 mod 10)=6 (filled with 16) ---- Disadvantage (it stuck in loop to find slot) so there is no guarantee to get slot once table filled 50 %.

**Pros :** No extra space required, Primary clustering resolved

**Cons :** Search O(n), Secondary clustering, finding slot

**2.3. Double Hashing :**

**h1(k) = K mod 11**

**h2(k) = 8 - (K mod 8)** (here 8 could be any value, should be less than hash fun value 11, 8 and 11 should be relatively prime)

**(h1(k) + i.h2(k)) mod 11**(this should be prime)

**Ex :** 20,34,45,70,56 and hash fun is K mod 11.

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 |
|  | 34 |  | 56 | 45 |  | 70 |  |  | 20 |  |

(h1(k) + i.h2(k)) mod 11 = (h1(45) + 1 \* (8-(45 mod 8))) mod 11 **=** (1 + 3) mod 11 = 4

(h1(k) + i.h2(k)) mod 11 = (h1(70) + 1 \* (8-(70 mod 8))) mod 11 = (4 + 2) mod 11 = 6

(h1(k) + i.h2(k)) mod 11 = (h1(56) + 1 \* (8-(56 mod 8))) mod 11 = (1 + 8) mod 11 = 9 (filled with 20), so i=2 (1+ 2\*8) mod 11 = 6 (filled with 70),

so i=3 (1+3\*8) mod 11 = 3

**Pros :** No extra space required, Primary and secondary clustering resolved

**Cons :** search time : good case O(1) worst case O(n)