

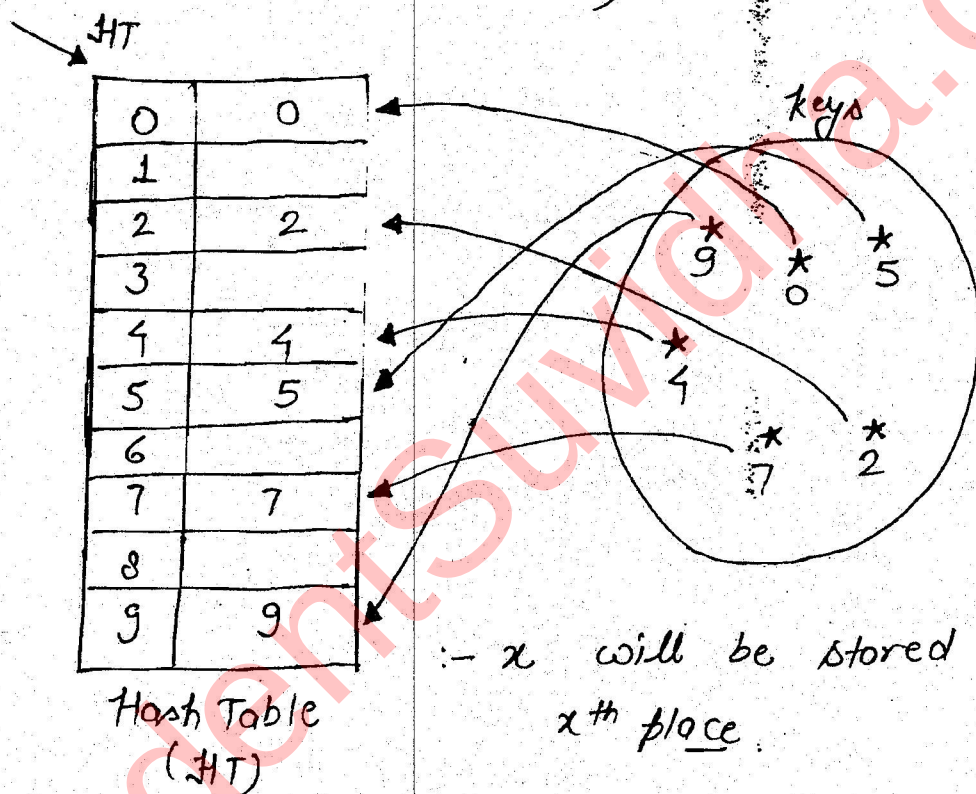
Dated
30/09/2017

*** # "Hashing" :-

- i) Searching Technique
- ii) worst case Searching Time = $O(1)$

Direct Address Table (DAT)

$m=10$ (0-9)



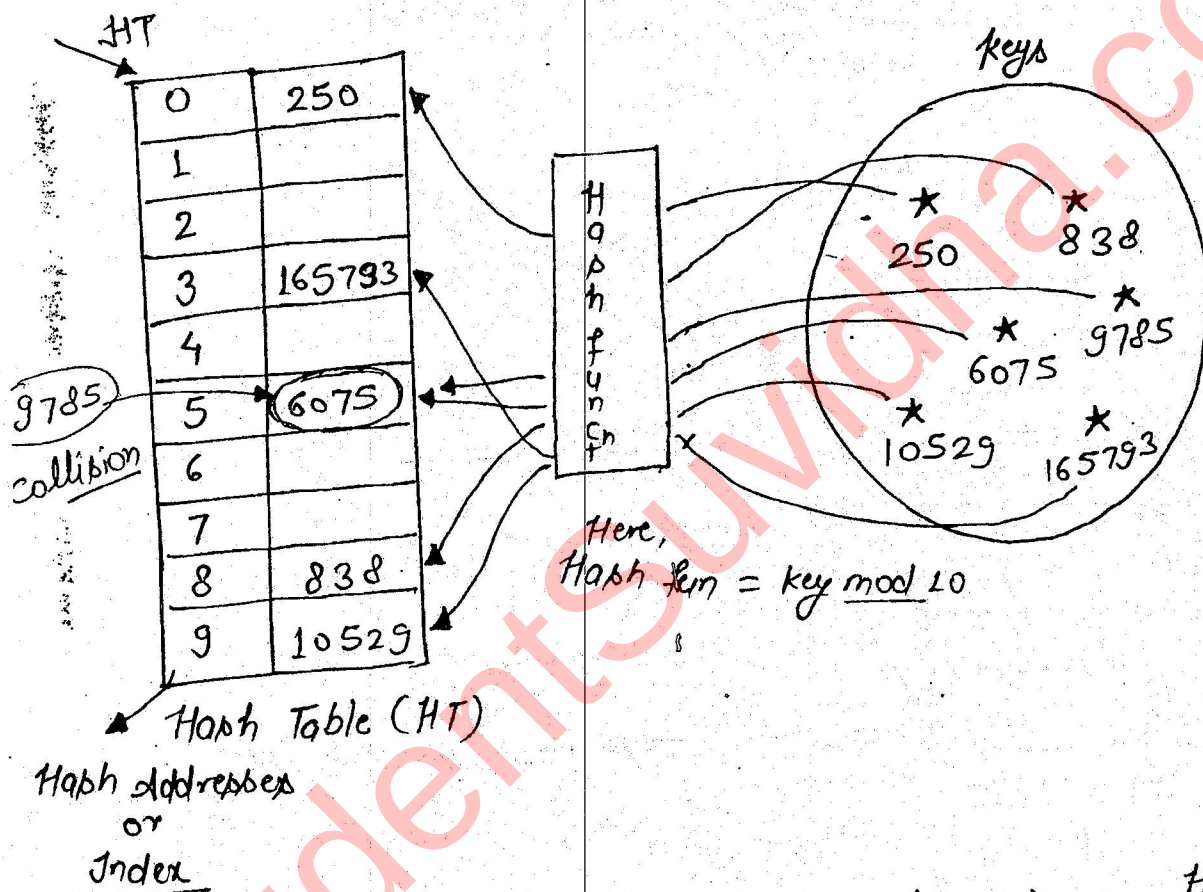
- * i) In Direct Address Table, key itself is the address without any calculation.
- * ii) ~~worst~~ Searching time = $O(1)$ { Every case }.
- * iii) The Drawback with the Direct Address Table is Even though no. of keys very less but one of the key very large (2^{1000}) then there is a need of Hash Table of size 2^{1000} approx.

To store key 2^{1000} , table of size 2^{1000} (approx) required, Huge space wastage.
 - To Eliminate above Drawback, we are going to the 'Hash functions'.

< 'Hash function' > -

$m = 10$ (0-9)

$m = \text{Hash table size}$.



< 'collision' > - If two keys contain same Hash - address, this is called as collision

$$\begin{aligned} 6075 \text{ mod } 10 &= 5 \\ 9785 \text{ mod } 10 &= 5 \end{aligned} \quad \left. \vphantom{\begin{aligned} 6075 \text{ mod } 10 &= 5 \\ 9785 \text{ mod } 10 &= 5 \end{aligned}} \right\} \text{Collision (same index)}$$

Types of Hash functions :-

- i) Division modulo method
- ii) Digit Extraction Extraction method
- iii) mid square method
- iv) folding method
 - (i) fold Boundary
 - (ii) fold shifting

i) Division modulo method :-

Ex: ① :- $m = 8 \text{ (or)} (2^3)$

$$\text{Hash fn (key)} \rightarrow hf(k) = \text{key mod } m \rightarrow \text{Hash table Size}$$

$$\text{key} = 1011010111000011$$

$$10110101110000\underbrace{011}_{\text{last 3 bits}} \text{ mod } 2^{\textcircled{3}} = \underbrace{011}_{\text{L.S.B 3 bits To Represent}}$$

$\therefore$ Remainder of mod 8 will be b/w (0 --- 7)
 $\therefore$ max is 7 which can be represented by 3 bits. so last 3 bits will be answer always of mod 8

Ex: ② - $m = 2^k$

$$\underbrace{1010100100010010011101100}_{k \text{ bits}} \text{ mod } 2^{\textcircled{k}} = \overline{\text{L.S.B } k \text{ bits}} = \underline{11101100}$$

< i) Don't Pick m value exactly Powers of 2 because if $m = 2^k$ answer is always L.S.B 'k bits' only

< ii) Pick 'm value' which is the Prime number. but it should not be ^{very} close to Powers of 2 like : $\underset{x}{61}, \underset{x}{511}, \underset{x}{1023}$ (Don't take), $\underset{\checkmark}{727}$

2) "Digit Extraction method" :-

Ex:- $m = 1000$ (0 - 999)

key = $\underset{1\ 2\ 3}{7\ 8\ 9}\ \underset{4\ 5\ 6}{4\ 5\ 6}\ \underset{7\ 8\ 9}{1\ 2\ 3}$

hf(key) = 963

{ which digits I have to extract will be 'given' in the question }

HT

0	
1	
2	
⋮	
963	789456123
⋮	
999	

{ with 1000 table size, 3 digits addresses will more majority (900) { 100-999 } so extracting 3 digits }

* This method also called as "Truncation method", as it only extract few digits and leave remaining.

* digits are specified to extract, so easy to get same key (more collisions)
Previous method better than this.

3) "Mid Square method" :-

$$m = 1000 \text{ (0-999)}$$

$$\text{key} = 84725$$

$$\text{hf}(\text{key}) = (84725)^2$$

$$= 7178325625$$

$$= \underline{832} \text{ (or) } \underline{325}$$

{ 3 digits because majority of addresses are 3 digit addresses }

// { first square then take mid }

:- In this method all digits are participating for square. Hard to get same key (less collision) more powerful than previous two

4) "Folding Method" :-

(i) "Fold Boundary method" :-

$$m = 1000 \text{ (0-999)}$$

$$\text{key} = \underline{789}123\underline{456}$$

$$\text{hf}(\text{key}) =$$

$$789$$

$$456$$

$$\underline{1245}$$

$$124$$

$$5$$

$$\underline{129}$$

* / taking 1st three & last 3 digits (boundary) * /

:- taking 3 digits because majority of addresses are 3 digits.

:- more than 3 digits then add last digits to 1st 3 digits.

ii) Fold Shifting :-
 $m = 1000$ (0 - 999)

key = (789)(123)(456)

$$\begin{array}{r}
 789 \\
 123 \\
 + 456 \\
 \hline
 1368 \\
 136 \\
 \hline
 144
 \end{array}$$

144 - 789 123 456

* In this method, all digits are participating
 so, getting same key will be difficult
 (Less collision).

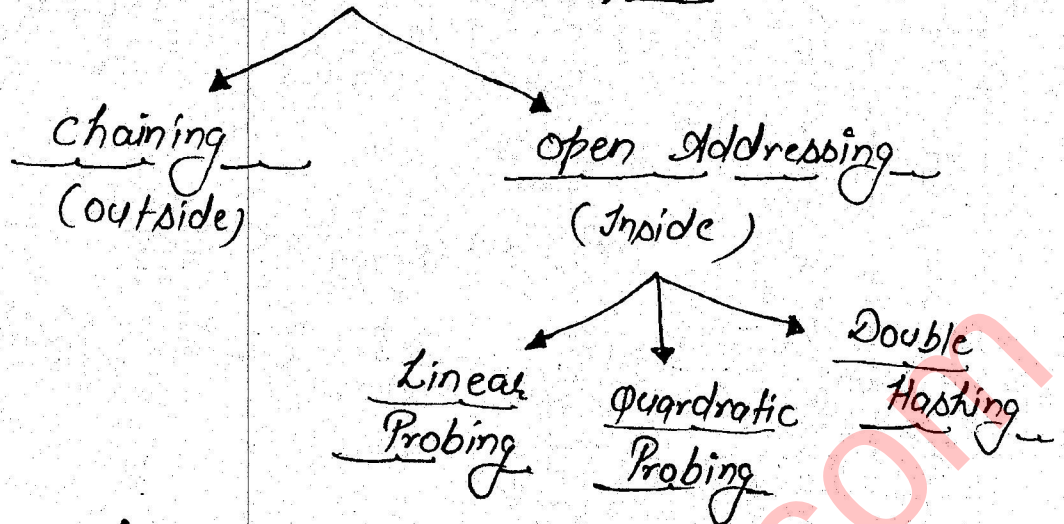
Note :- It can be assumed :-

:- mid Square > fold shifting > fold Boundary > Division module > Digit Extraction

try to
 /* take group of 3-3 digits
 add them, if
 more than 3 digits
 add again 3 digits group
 & remaining. */

/* taking 3 digits group,
 majority address 3 digit. */

"Collision Resolution Techniques" :-



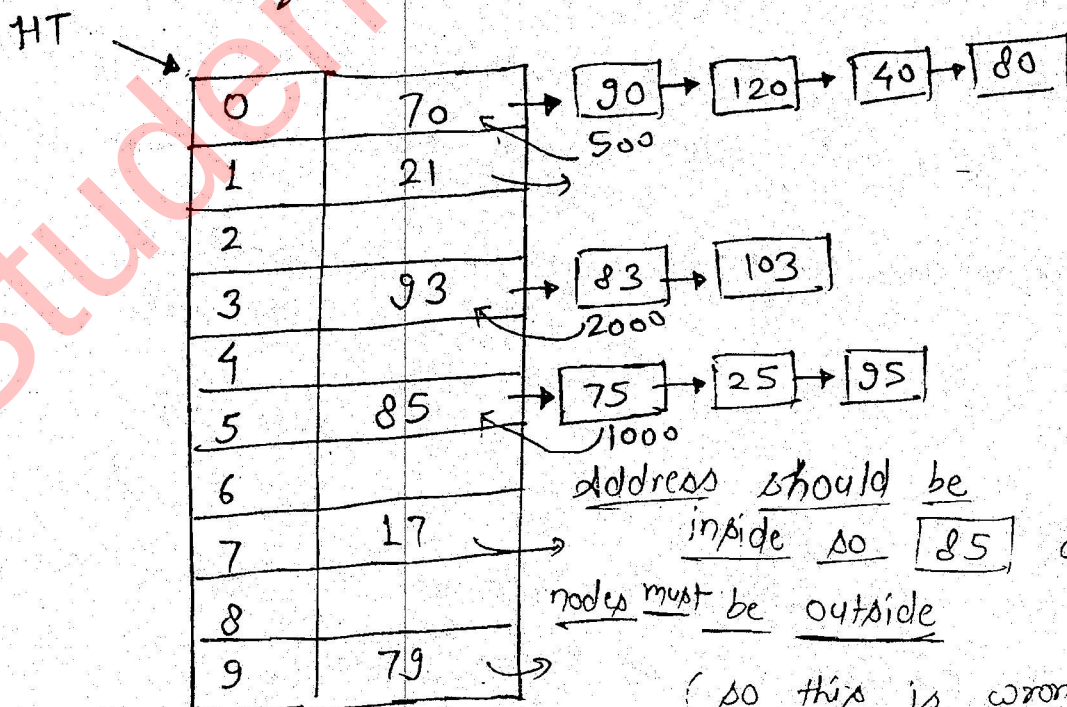
1.) "Chaining" :-

Ex: ①:- $m = 10 (0-9)$

$$hf(key) = key \bmod m$$

Keys = 70, 85, 93, 75, 90, 120, 79, 83, 25, 95,
40, 121, 17, 103, 80

Collision Resolution Technique = chaining.



Address should be inside so 85 also all other nodes must be outside

(so this is wrong way)
(Address & value can't store)

* v) insertion time :- $O(1)$: { Every case }

{ as we can insert at the first place

* vi) Deletion is by changing addresses, easier because deletion of one element will not trouble other.

2) "Open Addressing" :-

i) "Linear Probing" :- [Closed Hashing]
(covering all slots)

N = No. of keys

m = No. of slots

m slots = N keys

$$\frac{1\text{-slot}}{\text{avg (or)}} = \frac{N}{m} \text{ keys/slot}$$

$$\boxed{\text{Load factor } (\alpha) = \frac{N}{m}}$$

* for chaining :- $0 \leq \alpha \leq \infty$

* for open addressing :- $0 \leq \alpha \leq 1$

Ex: ① :- $m = 10$ (0-9)

$$hf(k) = k \bmod m$$

keys = 25, 70, 33, 35, 90, 46, 55, 80, 91

Collision Resolution = Linear Probing.
Technique

$$LP(\text{key}, i) = (hf(\text{key}) + i) \bmod m$$

Attempt
No. attempt.

i value changes
i.e. 0, 1, 2, ..., $m-1$
 $\forall i \in \{0, 1, 2, 3, \dots, m-1\}$

0	70
1	90
2	80
3	33
4	91
5	25
6	35
7	46
8	55
9	

$$LP(25,0) = (5+0) \bmod 10 = 5$$

$$LP(70,0) = (0+0) \bmod 10 = 0$$

$$LP(33,0) = (3+0) \bmod 10 = 3$$

$$LP(35,0) = (5+0) \bmod 10 = 5$$

$$(35,1) = (5+1) \bmod 10 = 6 \leftarrow 1-C \text{ collision}$$

$$LP(90,0) = (0+0) \bmod 10 = 0$$

$$(90,1) = (0+1) \bmod 10 = 1 \leftarrow 1 \text{ collision}$$

$$LP(46,0) = (6+0) \bmod 10 = 6$$

$$(46,1) = (6+1) \bmod 10 = 7 \leftarrow 1-C$$

$$LP(55,0) = (5+0) \bmod 10 = 5$$

$$(55,1) = (5+1) \bmod 10 = 6 \leftarrow 3-C$$

$$(55,2) = (5+2) \bmod 10 = 7$$

$$(55,3) = (5+3) \bmod 10 = 8$$

$$LP(80,0) = (0+0) \bmod 10 = 0$$

$$(80,1) = (0+1) \bmod 10 = 1$$

$$(80,2) = (0+2) \bmod 10 = 2 \leftarrow 2-C$$

$$LP(91,0) = (1+0) \bmod 10 = 1$$

$$(91,1) = (1+1) \bmod 10 = 2$$

$$(91,2) = (1+2) \bmod 10 = 3$$

$$(91,3) = (1+3) \bmod 10 = 4 \leftarrow 3-C$$

Total collisions :- 11-C

Note :-

If collision there, order of keys matters
If No collision on keys, they can be swapped.

* i) we are probing (searching) one slot after another (or one by one).

ii) we are not ~~pro~~ wasting space unnecessarily outside in the form of link-list.

iii) :- Searching Time = $O(n)$ or $O(m)$ (worst case)
also in open addr. $m > n$ keys

:- (Insertion time) = $O(1)$ Best case
name & also

:- Deletion time = $O(n)$ } worst case
= $O(1)$ } Best case

* (iv)

Deletion difficult because deletion of one Element will create trouble to other elements. but we can manage with the special symbol '\$' if more '\$' then 'ReHash' again.

2.) "Quadratic Probing" :-

$$m = 10 \text{ (0-9)}$$

$$h(k) = k \bmod m$$

$$\text{keys} = 25, 70, 33, 35, 90, 46, 55, 80, 91$$

Collision Technique = Quadratic Probing.

$$QP(\text{key}, i) = (h(\text{key}) + c_1 \cdot i + c_2 \cdot i^2) \bmod m$$

$$\forall i \in \{0, 1, 2, \dots, m-1\}$$

:- c_1 & c_2 are constants will be given in question

Let's take $c_1 = 1$ & $c_2 = 1$

0	70
1	55
2	90
3	80 33
4	
5	25
6	46
7	35
8	
9	

$$\phi P(25,0) = (5 + 1 \times 0 + 1 \times 0^2) \bmod 10 = 5$$

$$\phi P(70,0) = (0 + 1 \times 0 + 1 \times 0^2) \bmod 10 = 0$$

$$\phi P(33,0) = (3 + 1 \times 0 + 1 \times 0^2) \bmod 10 = 3$$

$$\phi P(35,0) = (5 + 1 \times 0 + 1 \times 0^2) \bmod 10 = 5$$

$$1C \left\{ \begin{aligned} (35,1) &= (5 + 1 \times 1 + 1 \times 1^2) \bmod 10 = 7 \end{aligned} \right.$$

$$\phi P(90,0) = (0 + 1 \times 0 + 1 \times 0^2) \bmod 10 = 0$$

$$1C \left\{ \begin{aligned} (90,1) &= (0 + 1 \times 1 + 1 \times 1^2) \bmod 10 = 2 \end{aligned} \right.$$

$$\phi P(46,0) = (6 + 1 \times 0 + 1 \times 0^2) \bmod 10 = 6$$

$$\phi P(55,0) = (5 + 1 \times 0 + 1 \times 0^2) \bmod 10 = 5$$

$$2C \left\{ \begin{aligned} (55,1) &= (5 + 1 \times 1 + 1 \times 1^2) \bmod 10 = 7 \\ (55,2) &= (5 + 2 \times 2 + 1 \times 2^2) \bmod 10 = 11 \bmod 10 = 1 \end{aligned} \right.$$

$$\phi P(80,0) = (0 + 1 \times 0 + 1 \times 0^2) \bmod 10 = 0$$

$$10-C \left\{ \begin{aligned} &= (0 + 1 \times 1 + 1 \times 1^2) \bmod 10 = 2 \\ &= (0 + 1 \times 2 + 1 \times 2^2) \bmod 10 = 6 \\ &= (0 + 1 \times 3 + 1 \times 3^2) \bmod 10 = 2 \\ &= (0 + 1 \times 4 + 1 \times 4^2) \bmod 10 = 0 \\ &= (0 + 1 \times 5 + 1 \times 5^2) \bmod 10 = 0 \\ &= (0 + 1 \times 6 + 1 \times 6^2) \bmod 10 = 2 \\ &= (0 + 1 \times 7 + 1 \times 7^2) \bmod 10 = 6 \\ &= (0 + 1 \times 8 + 1 \times 8^2) \bmod 10 = 2 \\ &= (0 + 1 \times 9 + 1 \times 9^2) \bmod 10 = 0 \end{aligned} \right.$$

max

$$DP(91, 0) = (1 + 1 \times 0 + 1 \times 0^2) \mod 10 = 1$$

$$\underbrace{\quad}_{10-C} \quad \left\{ \begin{array}{c} \vdots \\ \vdots \end{array} \right.$$

$$\text{Total Collisions} = 24$$

:- 91, & 80 can't go inside table, even space is available

* i) space is not waiting in the form of Linklist

* ii) $\underbrace{\text{Searching time}}_A = \begin{array}{l} O(1) \text{ } \{ \text{B.C} \\ O(m) \text{ or } O(n) \text{ } \{ \text{worst case} \end{array}$

$\underbrace{\text{insertion time}}_B = \begin{array}{l} O(1) \text{ } \{ \text{Best case} \\ O(n) \text{ } \{ \text{w.c.} \end{array}$

$\underbrace{\text{Deletion time}}_C = \begin{array}{l} O(1) \text{ } \{ \text{B.C} \\ O(n) \text{ } \{ \text{w.c.} \end{array}$

* iii) Deletion of one Element will trouble other Element but we can manage with the special Symbol '\$' If more '\$' Refresh again.

3.) "Double Hashing" :-

[Two Hash functions hf_1 , hf_2 , these will be given in question].

Ex: ① :-

$$m = 10 \quad (0-9)$$

$$hf_1(k) = k \bmod m$$

$$\text{keys} = 25, 70, 33, 35, 90, 46, 55, 80, 91$$

C.R.T = Double Hashing
Collision Resolution Tech

$$DH(\text{key}, i) = (hf_1(\text{key}) + i \cdot hf_2(k)) \bmod m$$

$$\therefore hf_2(k) = (1 + (k \bmod (m-2)))$$

0	70 35
1	55
2	80
3	33
4	
5	25
6	90
7	46
8	
9	35

$$DH(25, 0) = (5 + 0 \cdot hf_2(k-2)) = 5$$

$$DH(70, 0) = (0 + 0 \cdot hf_2(k-2)) = 0$$

$$DH(33, 0) = (3 + 0 \cdot hf_2(k-2)) = 3$$

$$DH(35, 0) = (5 + 0 \cdot hf_2(k-2)) = 5$$

$$(35, 1) = (5 + 1 \cdot (1+3)) = 9$$

$$\therefore \{ (35, 1) = (5 + 1 \cdot (1+3)) = 9$$

$$DH(90, 0) = (0 + 0 \cdot hf_2(k-2)) = 0$$

$$2-c \left\{ \begin{aligned} (90, 1) &= (0 + 1 \cdot (1+2)) \bmod 10 = 3 \\ DH(90, 2) &= (0 + (2 \cdot (1+2))) \bmod 10 = 6 \end{aligned} \right.$$

$$DH(46, 0) = (6 + 0) = 6$$

$$DH(46, 1) = (6 + 1 \cdot (1+6)) \bmod 10 = 3$$

$$3-C \left\{ \begin{aligned} (46, 2) &= (6 + 2 * (1 + 6)) \mod 10 = 0 \\ &= (6 + 3 * (1 + 6)) \mod 10 = 7 \end{aligned} \right.$$

$$DH(55, 0) = 5$$

$$2-C \left\{ \begin{aligned} (55, 1) &= (5 + 1 * (1 + 7)) \mod 10 = 3 \\ &= (5 + 2 * 8) \mod 10 = 1 \end{aligned} \right.$$

$$DH(80, 0) = 0$$

$$2-C \left\{ \begin{aligned} (80, 1) &= (0 + 1 * (1 + 0)) \mod 10 = 1 \\ (80, 2) &= (0 + 2 * (1)) = 2 \end{aligned} \right.$$

$$DH(91, 0) = 1$$

$$10-C \left\{ \begin{aligned} (91, 9) & \end{aligned} \right.$$

$$\text{Total collisions} = 20 \underline{C}$$

91 can't go inside, even space available

* i) Space not wasting in form of LinkedList outside

* ii) $\left. \begin{array}{l} \text{searching time} \\ \text{insertion time} \\ \text{deletion time} \end{array} \right\} = \begin{array}{l} O(1) \text{ Best case} \\ O(n) \text{ worst case} \end{array}$

* iii) Deletion will trouble, but we can manage with '\$' & more '\$' rehash again.

Que :-

Linear Probing

$$m = 10 \text{ (0-9)}$$

$$h(k) = k \bmod m$$

$$\text{keys} = 53, 60, 71, 92, 80, 90$$

0	60
1	71
2	92
3	53
4	→ 80
5	90
6	
7	
8	
9	

Primary clustering -

$$LP(53, 0) = (3 + 0) \bmod 10 = 3$$

$$(60, 0) = (0 + 0) = 0$$

$$(71, 0) = (1 + 0) \bmod 10 = 1$$

$$(92, 0) = (2 + 0) \bmod 10 = 2$$

$$LP(80, 0) = 0$$

$$(80, 1) = 1$$

$$(80, 2) = 2$$

$$(80, 3) = 3$$

$$(80, 4) = (0 + 4) \bmod 10 = 4$$

$$LP(90, 0) = 0$$

$$(90, 1) = (0 + 1) = 1$$

$$(90, 2) = (0 + 2) \bmod 10 = 2$$

$$(90, 3) = (0 + 3) \bmod 10 = 3$$

$$(90, 4) = (0 + 4) \bmod 10 = 4$$

$$(90, 5) = (0 + 5) \bmod 10 = 5$$

Primary clustering:-

To insert 90 after the 80 have to do the same work and have to cross the (group of keys) a cluster, so this is called Primary clustering due to this avg searching time increases.

Primary clustering :-

i) If two keys contain, same starting Hash-Address they both will follow same path unnecessarily in linear manner because of this average searching time will increase, this problem is known as Primary clustering.

* ii) Linear Probing having the disadvantage of Primary-clustering.

* iii) Searching time = $O(1)$ ^{Best case} ~~worst case~~
 $O(m)$ ^{worst case}

$$\text{Avg case : } 1 + 2 + 3 + \dots + m$$

$$= \frac{m(m+1)}{2} \Rightarrow \frac{m+1}{2} \Rightarrow \underline{O(m)}$$

Que :

Quadratic Probing :-

$$m = 10 \quad (0-9)$$

$$h(k) = k \bmod m$$

$$\text{keys} = 53, 60, 71, 92, 80, 90 \quad C_1 = 1, C_2 = 1$$

0	60
1	71
2	92
3	53
4	
5	
6	80
7	
8	
9	

$$\phi(80, 0) = (0 + 1 \times 0 + 1 \times 0^2) = 0$$

$$\phi(80, 1) = (0 + 1 \times 1^2) = 1$$

$$\phi(80, 2) = (0 + 2 \times 1 + 1 \times 2^2) = 6$$

$$\phi(90, 0) = 0$$

$$\phi(90, 1) = (0 + 1 \times 1 + 1 \times 1^2) = 2$$

$$\phi(90, 2) = (0 + 1 \times 2 + 1 \times 2^2) = 6$$

$$\phi(90, 3) = (0 + 1 \times 3 + 1 \times 3^2) \bmod 10 = 2$$

$$\phi(90, 4) = (0 + 1 \times 4 + 1 \times 4^2) \bmod 10 = 0$$

same work

Secondary Clustering :-

i) If two keys contain, same starting Hash Address they both will follow same path unnecessarily in quadratic manner because of this reason avg searching time will increase (It is less than the previous one) this problem is known as 'Secondary clustering'.

ii) * Quadratic Probing having the Disadvantage of Secondary clustering.

iii) Searching time = $O(1)$ Best case
= $O(m)$ worst case
= $O(m)$ Avg case (only constant factor will decrease)

Que :- Double Hashing :-

$$m = 10 \quad (0-9)$$

$$hf_1(k) = k \bmod m$$

$$keys = \cancel{55} 53, 60, 71, 92, 80, 90$$

$$CRT = D.H. \text{ (Double hashing)}$$

$$hf_2(k) = 1 + (k \bmod (m-2))$$

$$DH (key, i) = (hf_1 + i * hf_2) \bmod m$$

$$\underline{G.H.}, \underline{G.H.}$$

0	60
1	71
2	92
3	53
4	80
5	
6	90
7	
8	
9	

$$DH(80, 0) = (0 + 0 \times (0+1)) = 0$$

$$DH(80, 1) = (0 + 1 \times 1) = 1$$

$$(80, 2) = (0 + 2 \times 1) = 2$$

$$(80, 3) = (0 + 3 \times 1) = 3$$

$$(80, 4) = (0 + 4 \times 1) = 4$$

multiple
of 1

$$DH(90, 0) = (0 + 0 \times (1+2)) = 0$$

$$(90, 1) = (0 + 1 \times (1+2)) = 3$$

$$(90, 2) = (0 + 2 \times (1+2)) = 6$$

multiple
of 3

No
same
work
(different work)

different
path

$$DH(100, 0) = (0 + 0 \times (1+4)) = 0$$

$$(100, 1) = (0 + 1 \times (1+4)) = 5$$

$$(0 + 1 \times (1+4)) = 5$$

multiple
by 5

(Double Hashing)
Searching time :-

Avg case :- All are going for different path
(less redundancy)

No search time as total will be $\approx m$

$$\text{Avg} = \frac{m}{m} = \underline{O(1)}$$

* Note

* i) Double Hashing Avg case searching time = $O(1)$
because Every key will follow different
path because of 2nd Hash function.

* ii) In Double Hashing, 'Secondary clustering'
problem is eliminated.

:- using 'Perfect Hashing', we can get the
worst case searching time = $\underline{O(1)}$