* Chained Hash Search (T, K)

Search for an element with Key K in linked list $T[h(K)]$

* Chained Hash Delete (T, x)

Delete (x) from the list $T(h(key[x]))$

Worst Case searching time is $\Theta(n)$ + time of hash fn computation, which is no better than single linked list. But Hash tables are not used for their worst case performance but due to their average case performance.

Average Case performance of hashing depends on how well the hash function (h) distributes the set of keys to be stored among the m slots on an average.

B Open Addressing:

Add new record (R) with key (K) in our file (F), But h(K) is already occupied → Collision

Load factor: n — no. of keys in K
(no. of records in F)

m — no. of hash address in (L)

$$\lambda = \frac{n}{m} \quad \text{(load factor)}$$

⇒ Efficiency of hash fn with collision resolution is computed as

a) Average no. of probes (Key Comparisons)
- ↳ Successful Search $S(\lambda)$
- ↳ Unsuccessful Search $U(\lambda)$

Both values are computed in terms of (λ) the load factor.

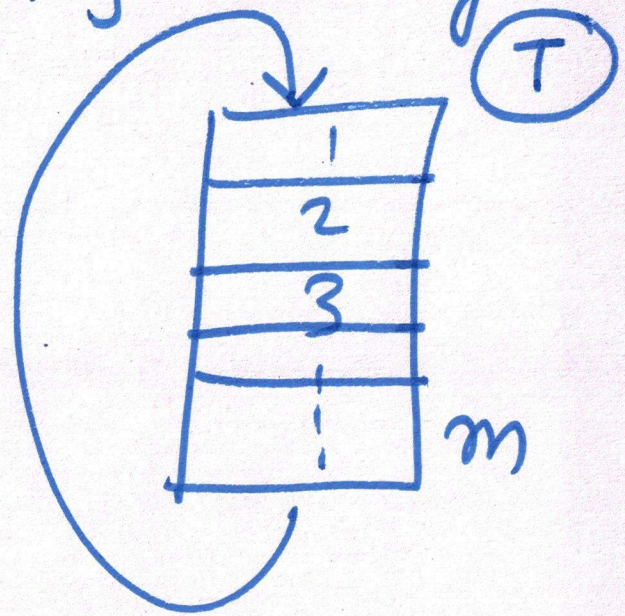
Linear Probing:

adding record (R) with key (K) into table (T)
Such that $H(K) = h$ (If (h) is filled Collision occured)

Insertion

→ If Collision occurs, assign (R) to first available location following T(h)

assuming [T] ≡ as Cyclic



Searching To Search a record (R) in Table (T)

↳ search linearly the locations $T[h]$, $T(h+1)$

until finding R (Successful) **OR**

meeting an empty location unsuccessful

$$S(\lambda) = \frac{1}{2}\left(1 + \frac{1}{1-\lambda}\right)$$

$$U(\lambda) = \frac{1}{2}\left(1 + \frac{1}{(1-\lambda)^2}\right)$$

Table (T) = 11 memory location (m)

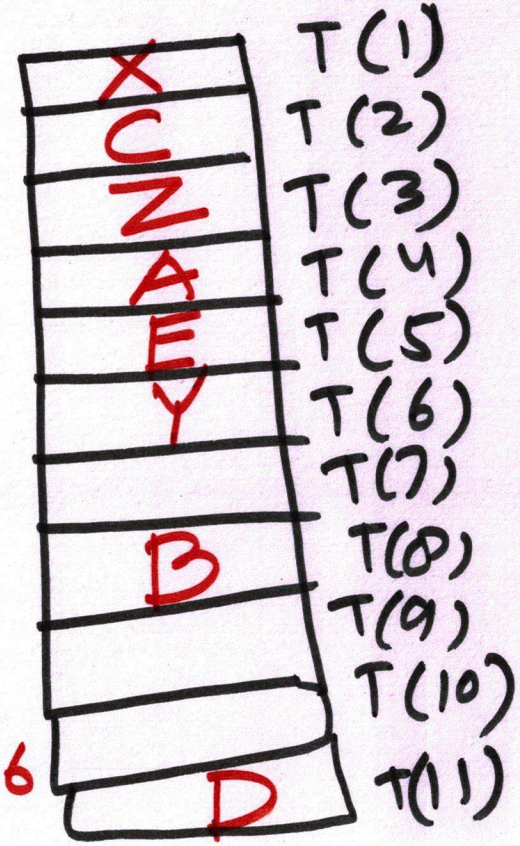
T[1], T[2] ——— T[11]

F = 8 Records (n)

Record →	A	B	C	D	E	X	Y	Z
H(k)	4	8	2	11	4	11	5	1

given by some hash fn.

- Records are entered into [T] in above order



no. of probes to find each of 8 records

$$S = \frac{1+1+1+1+2+2+2+3}{8} = \frac{13}{8} = 1.6$$

no. of probes to get empty location for each 11 location

$$U = \frac{7+6+5+4+3+2+1+2+1+1+8}{11} = \frac{40}{11} = 3.6$$

Quadratic Probing

*) Record (R) — Key(K) → Hash add $H(K) = h$

Instead of searching location with address

↳ $h, h+1, h+2$ — — —
(linear probing)

Search location

↳ $h, h+1, h+4, h+9, h+16$
(Quadratic probing