



# Estructura de Datos

Maria C. Torres

# Maria C. Torres

Ing. Electrónica (UNAL)

M.E. Ing. Eléctrica (UPRM)

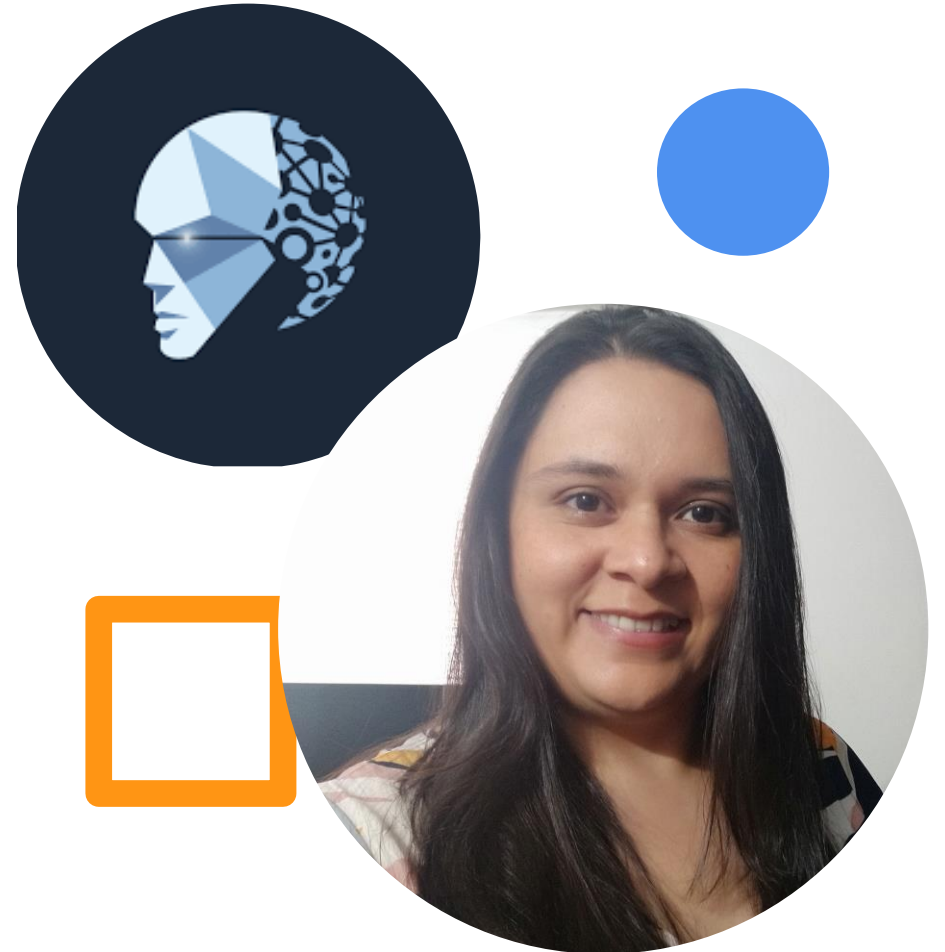
Ph.D. Ciencias e Ingeniería de la Computación y la Información (UPRM)


Profesora asociada

Dpto. Ciencias de la Computación y la Decisión

[mctorresm@unal.edu.co](mailto:mctorresm@unal.edu.co)

HORARIO DE ATENCIÓN: Martes 10:00 am a 12:00 m – Oficina 313 M8A





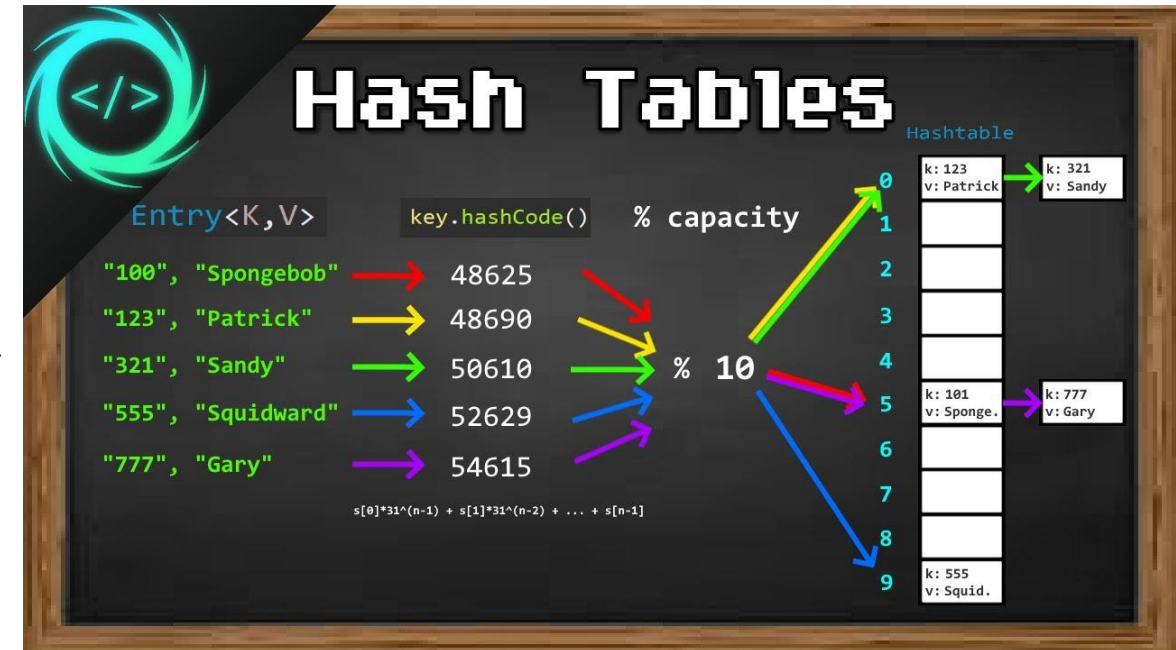
# Contenido del Curso

- ☐ Introducción: revisión fundamentos y POO
- ☐ Análisis de complejidad
- ☐ Arreglos
- ☐ Listas enlazadas
- ☐ Pilas y colas
- ☐ Heap
- ☐ Árboles binarios
- ☐ **Tablas hash**
- ☐ Grafos

# Tablas de dispersión

- ❑ Hash Tables
- ❑ Estructura de datos efectiva para implementar un diccionario que soporta las operaciones de INSERT, DELETE, y SEARCH.
- ❑ Un diccionario almacena datos asociados a una clave:

key	Data
-----	------

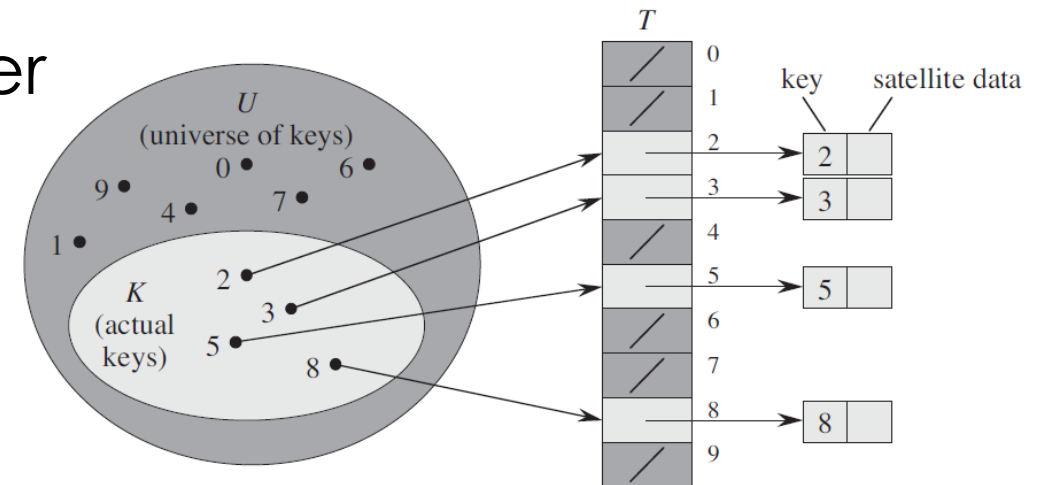


# Tablas de direccionamiento directo

- ❑ Suponga un arreglo donde  $i = \text{key}$
- ❑ Técnica simple que funciona bien cuando el universo de claves es pequeño
- ❑ i.e. La dimensión del arreglo debe ser igual a la máxima clave
- ❑ Asumiendo que ningún par de elementos tiene la misma clave.
- ❑ Se puede representar este conjunto dinámico como un arreglo

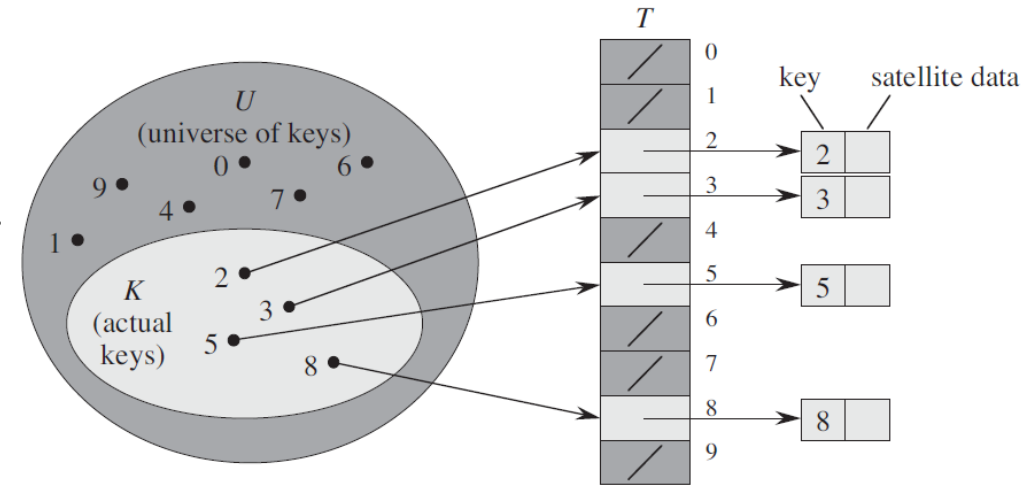
$T[0 \dots m-1]$

en el cual cada posición corresponde a una clave.



# Tablas de direccionamiento directo

- ❑ Cada operación tiene un tiempo de ejecución  $O(1)$ .
- ❑ La principal desventaja del direccionamiento directo es que, si el universo  $U$  es muy grande, almacenar la información es imposible, o si pocas claves son usadas, entonces la mayoría de espacio se desperdicia.



DIRECT-ADDRESS-SEARCH( $T, k$ )

1 **return**  $T[k]$

DIRECT-ADDRESS-INSERT( $T, x$ )

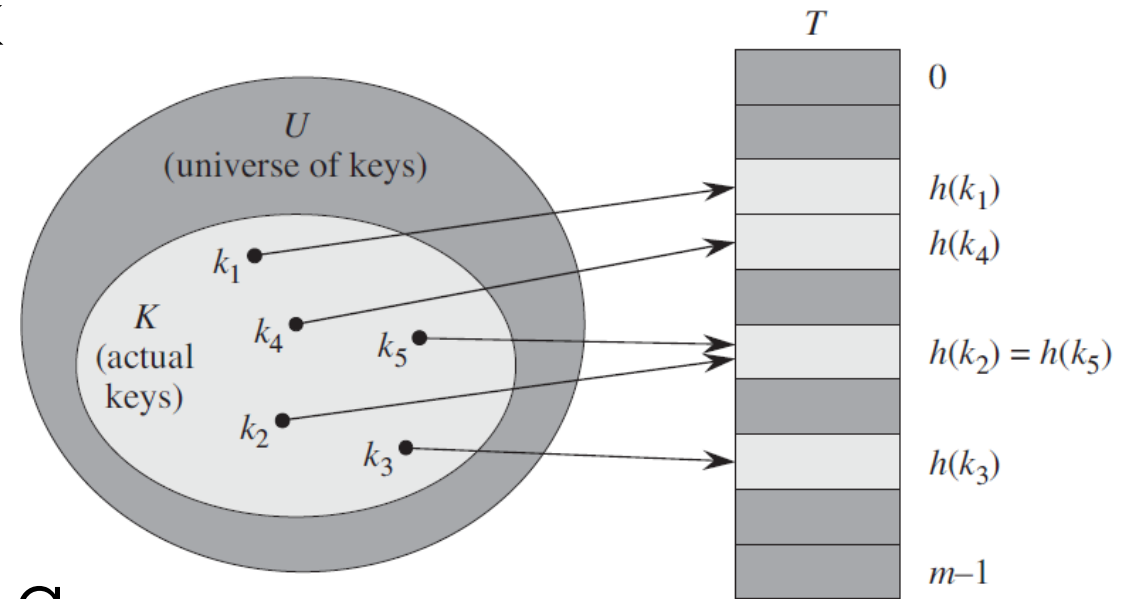
1  $T[x.key] = x$

DIRECT-ADDRESS-DELETE( $T, x$ )

1  $T[x.key] = \text{NIL}$

# Tablas de dispersión

- ❑ En una tabla de direccionamiento directo, un elemento con clave  $k$  se almacena en la posición  $k$ .
- ❑ En una tabla de dispersión, un elemento con clave  $k$  se almacena en la posición  $h(k)$ .
- ❑  $h(k)$  se denomina la **función de dispersión** (hash function).
- ❑  $h$  mapea el universo  $U$  de claves a las posiciones en una tabla de dispersión  $T[0...m-1]$ , donde  $m$  es típicamente mucho más pequeño que el tamaño del universo  $U$ .

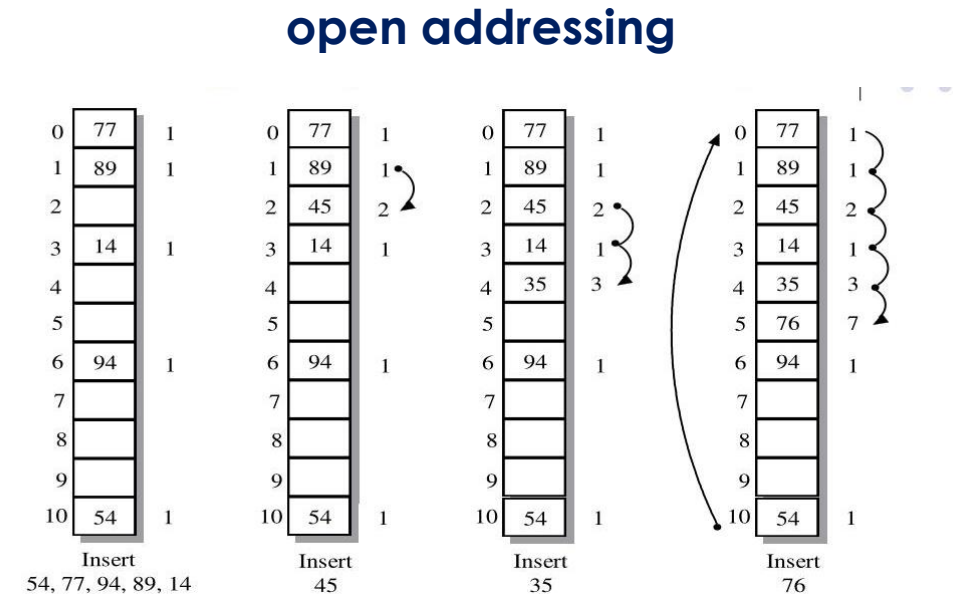
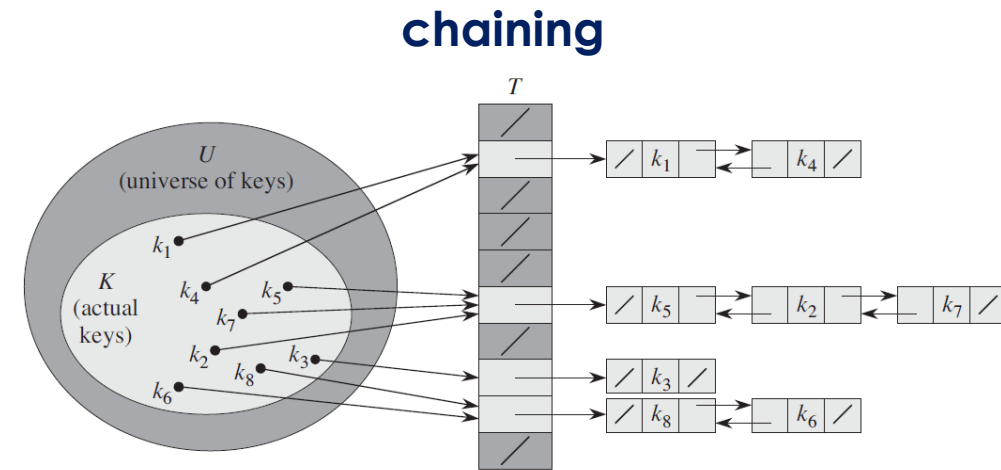


# Tablas de dispersión

❑ Esta asignación puede presentar problemas cuando dos claves se asignan a la misma posición en la tabla. Esta situación se denomina una **colisión**.

❑ Existen diferentes técnicas para solucionar colisiones en tablas de dispersión.

- Solución de colisiones por encadenamiento (**chaining**)
- Direcccionamiento abierto (**open addressing**)

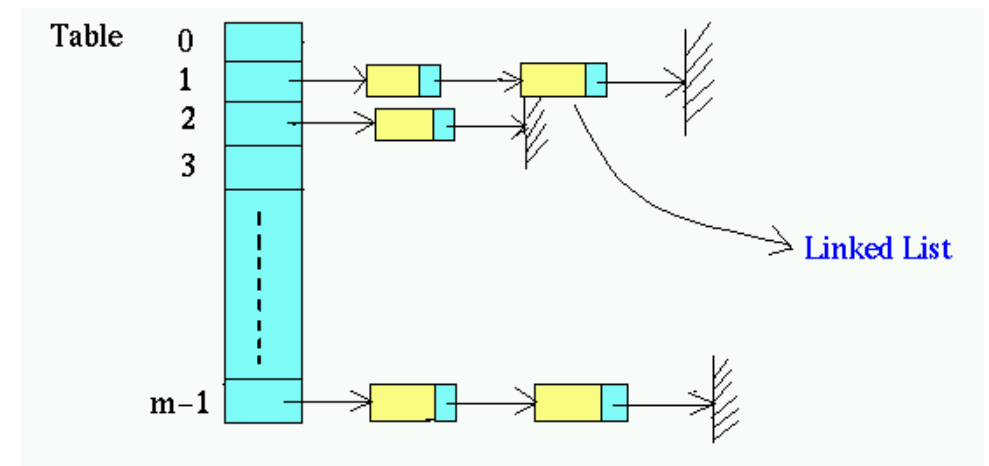
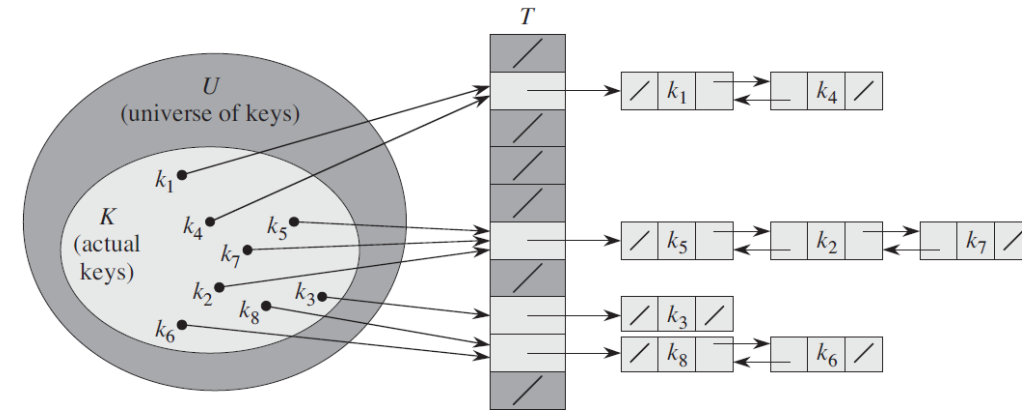




# Encadenamiento

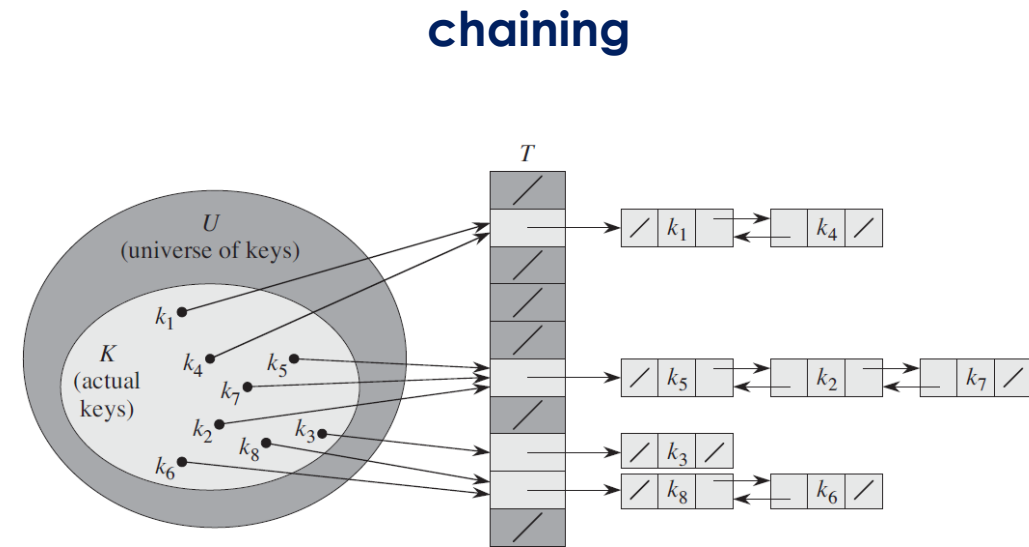
- ❑ La solución de colisiones por encadenamiento consiste en ubicar todos los elementos que se direccionan a la misma posición dentro de una lista enlazada.
- ❑ Cada posición  $j$  contiene un apuntador a la cabeza de la lista que almacena todos los elementos que se direccionan a la posición  $j$ , si no hay ningún elemento, la posición  $j$  contiene NULL.

## chaining



# Encadenamiento

- ❑ La operación INSERT y DELETE tiene un tiempo de ejecución  $O(1)$ .
- ❑ La operación INSERT asume que el elemento a insertar no se encuentra en la lista.



CHAINED-HASH-INSERT( $T, x$ )

1 insert  $x$  at the head of list  $T[h(x.key)]$

CHAINED-HASH-SEARCH( $T, k$ )

1 search for an element with key  $k$  in list  $T[h(k)]$

CHAINED-HASH-DELETE( $T, x$ )

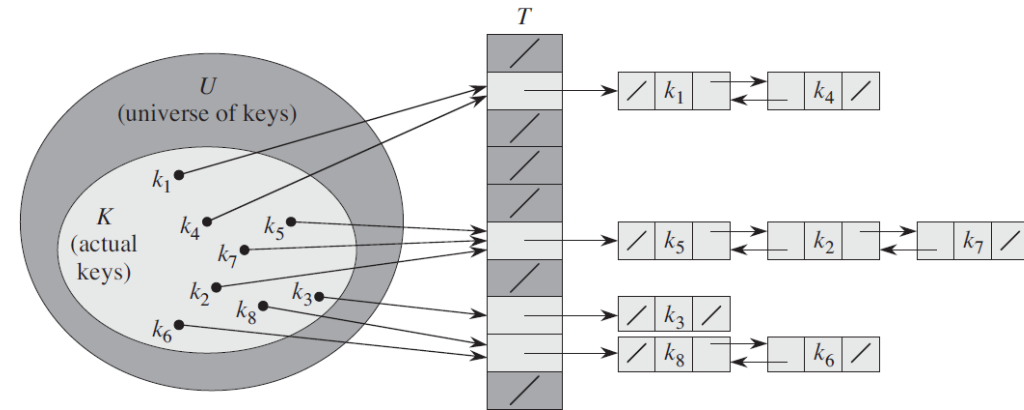
1 delete  $x$  from the list  $T[h(x.key)]$

# Encadenamiento

Búsqueda de un elemento:

- ❑ En el peor de los casos, todas las claves  $n$  se direccionan a la misma posición de la tabla, creando una lista de longitud  $n$ .
- ❑ Así, en el peor de los casos la búsqueda de un elemento con clave  $k$  en la tabla  $T$  es  $O(n)$  más el tiempo en calcular la función de dispersión.

## chaining



CHAINED-HASH-INSERT( $T, x$ )

1 insert  $x$  at the head of list  $T[h(x.key)]$

CHAINED-HASH-SEARCH( $T, k$ )

1 search for an element with key  $k$  in list  $T[h(k)]$

CHAINED-HASH-DELETE( $T, x$ )

1 delete  $x$  from the list  $T[h(x.key)]$

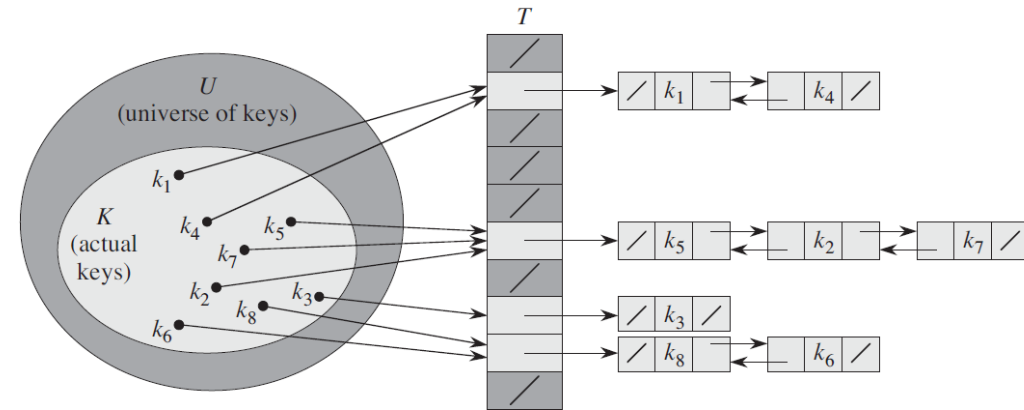
# Encadenamiento

Búsqueda de un elemento:

❑ Dada una tabla de dispersión  $T$  con  $m$  posiciones que almacenan  $n$  elementos, definimos el **factor de carga** para  $T$  como  $n/m$ , esto es el promedio de elementos almacenado en una lista.

❑ En el análisis del caso promedio (average-case) el desempeño de la búsqueda depende de cuan distribuidas quedan las claves a lo largo de las  $m$  posiciones.

## chaining



CHAINED-HASH-INSERT( $T, x$ )

1 insert  $x$  at the head of list  $T[h(x.key)]$

CHAINED-HASH-SEARCH( $T, k$ )

1 search for an element with key  $k$  in list  $T[h(k)]$

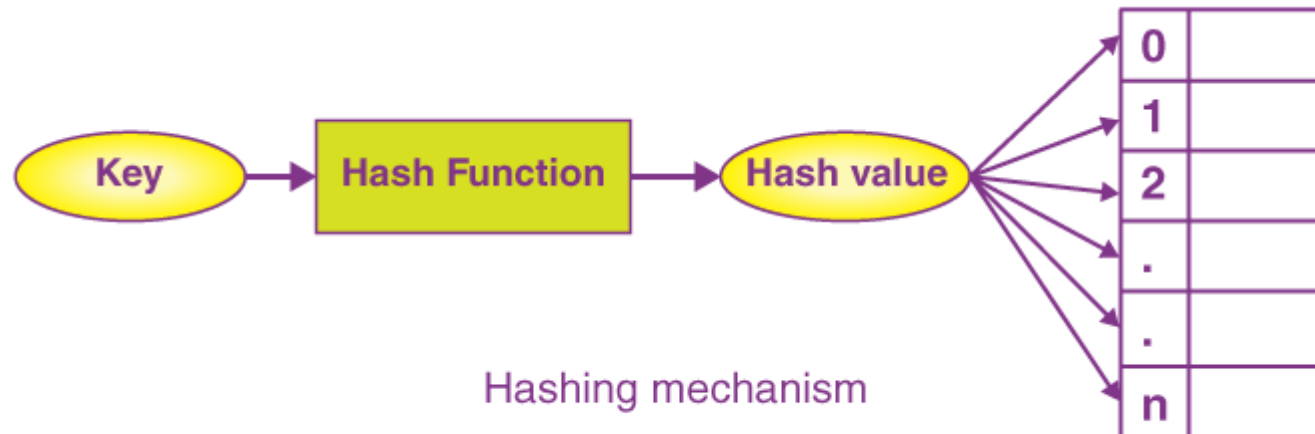
CHAINED-HASH-DELETE( $T, x$ )

1 delete  $x$  from the list  $T[h(x.key)]$

# Tablas de dispersión

¿Que hace una buena función de dispersión?

- ❑ Una buena función de dispersión satisface que cada clave puede ser asignada a cualquiera de las  $m$  posiciones con la misma probabilidad.
- ❑ En las siguientes funciones de dispersión vamos a asumir que la clave pertenece al conjunto de números naturales  $\{0, 1, 2, \dots\}$ .



# Tablas de dispersión

## Método de la división

Asigna una clave  $k$  a la posición dada por el residuo de la división de  $k$  entre  $m$  :

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

Por ejemplo: si el tamaño de la tabla de dispersión es  $m=12$  y la clave  $k=100$ , entonces  $h(k)=4$

❑ Cuando se usa este método usualmente se evita que  $m$  sea una potencia de 2. Un número primo no muy cercano a una potencia de 2 es una buena elección.

# Tablas de dispersión

## Método de la multiplicación

Primero se multiplica la clave  $k$  por una constante  $A$  en el rango de  $(0, 1)$  y se extrae la parte decimal. Luego se multiplica este valor por  $m$  y se toma el número entero no superior (función piso):

$$h(k) = \lfloor m(kA \bmod 1) \rfloor$$

❑ Este método trabaja para cualquier valor de  $A$ , sin embargo trabaja mejor para algunos valores dependiendo de los datos. Un valor sugerido de  $A$  es:

$$A \approx (\sqrt{5} - 1)/2 = 0,6180339887....$$

# Tablas de dispersión

## Ejemplo

- Vamos a insertar en una table con  $m = 9$  las siguientes claves, empleando el método de división: 5, 28, 19, 15, 20, 33, 12, 17, 10



# Tablas de dispersión

## Ejemplo

- Claves: 5, 28, 19, 15, 20, 33, 12, 17, 10

0
1
2
3
4
5
6
7
8

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

# Tablas de dispersión

## Ejemplo

- Claves: 5, 28, 19, 15, 20, 33, 12, 17, 10

$$h(5) = 5 \bmod 9 = 5$$

0	
1	
2	
3	
4	
5	5 /
6	
7	
8	

# Tablas de dispersión

## Ejemplo

- Claves: 5, 28, 19, 15, 20, 33, 12, 17, 10

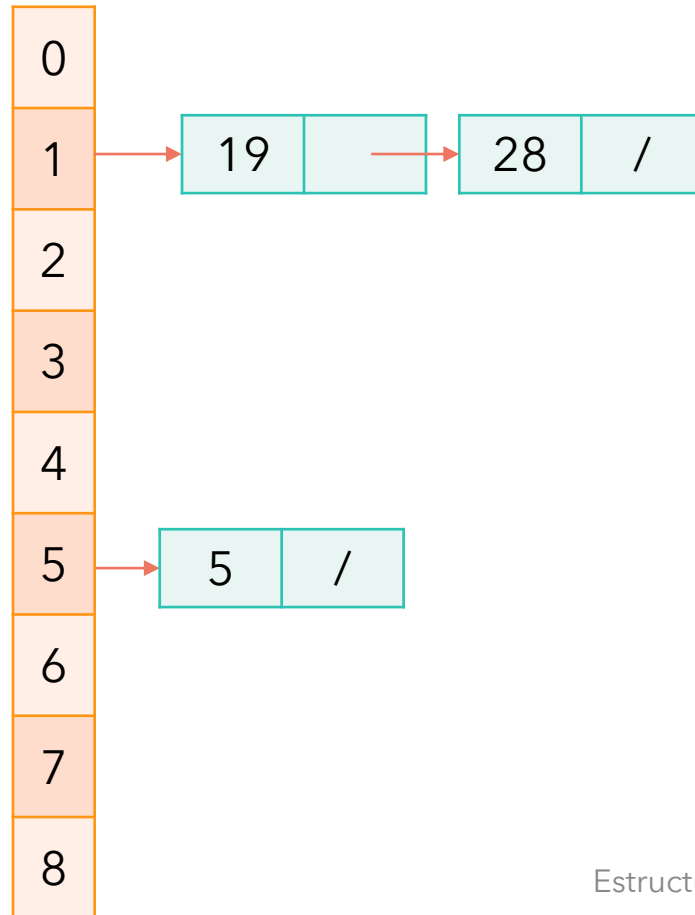
0	
1	→ 28 /
2	
3	
4	
5	→ 5 /
6	
7	
8	

$$h(28) = 28 \bmod 9 = 1$$

# Tablas de dispersión

## Ejemplo

- Claves: 5, 28, 19, 15, 20, 33, 12, 17, 10

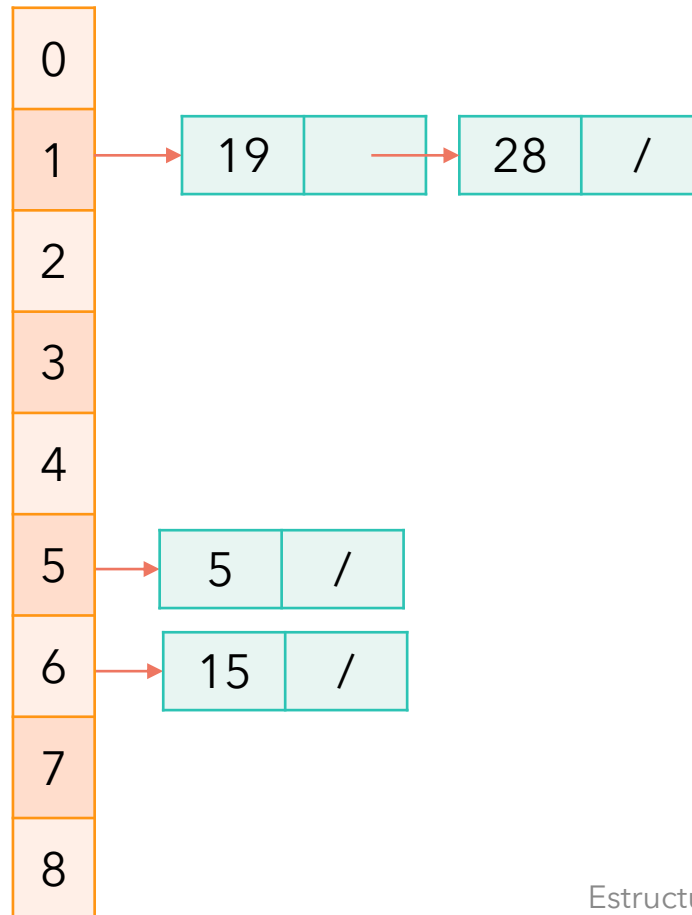


$$h(19) = 19 \bmod 9 = 1$$

# Tablas de dispersión

## Ejemplo

- Claves: 5, 28, 19, 15, 20, 33, 12, 17, 10

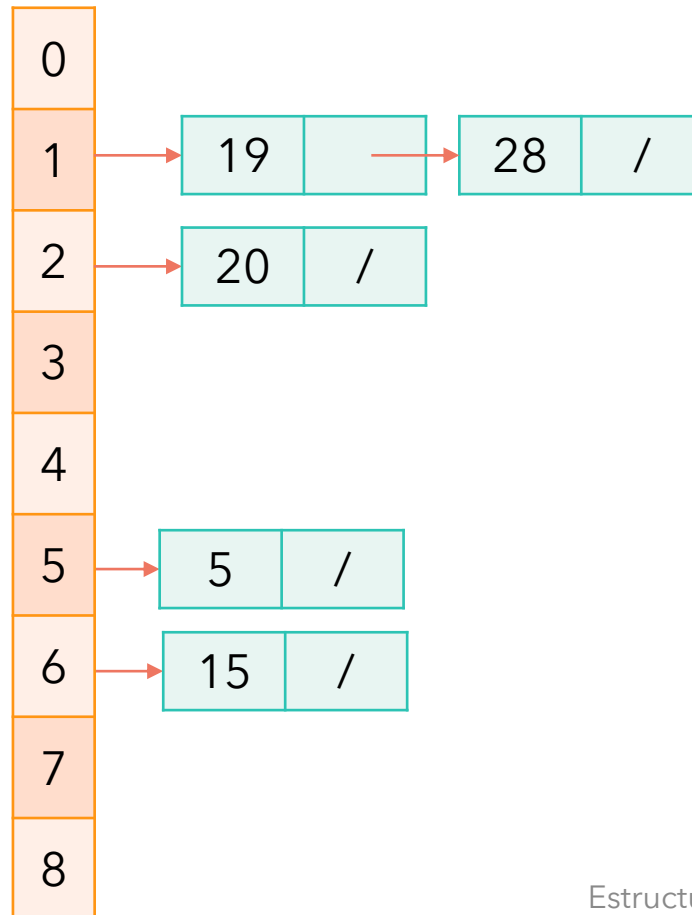


$$h(15) = 15 \bmod 9 = 6$$

# Tablas de dispersión

## Ejemplo

- Claves: 5, 28, 19, 15, 20, 33, 12, 17, 10

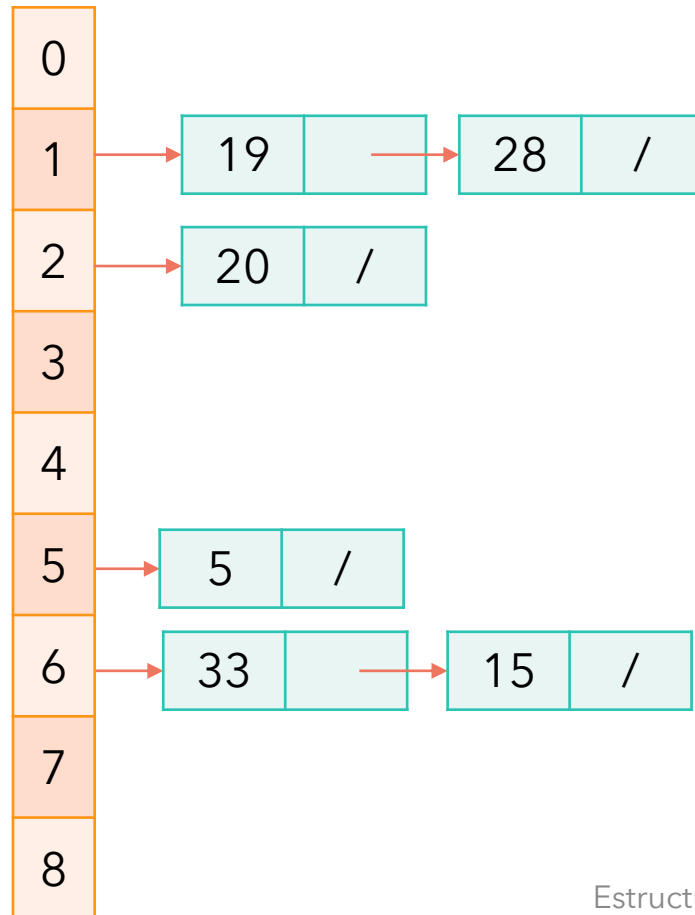


$$h(20) = 20 \bmod 9 = 2$$

# Tablas de dispersión

## Ejemplo

- Claves: 5, 28, 19, 15, 20, 33, 12, 17, 10

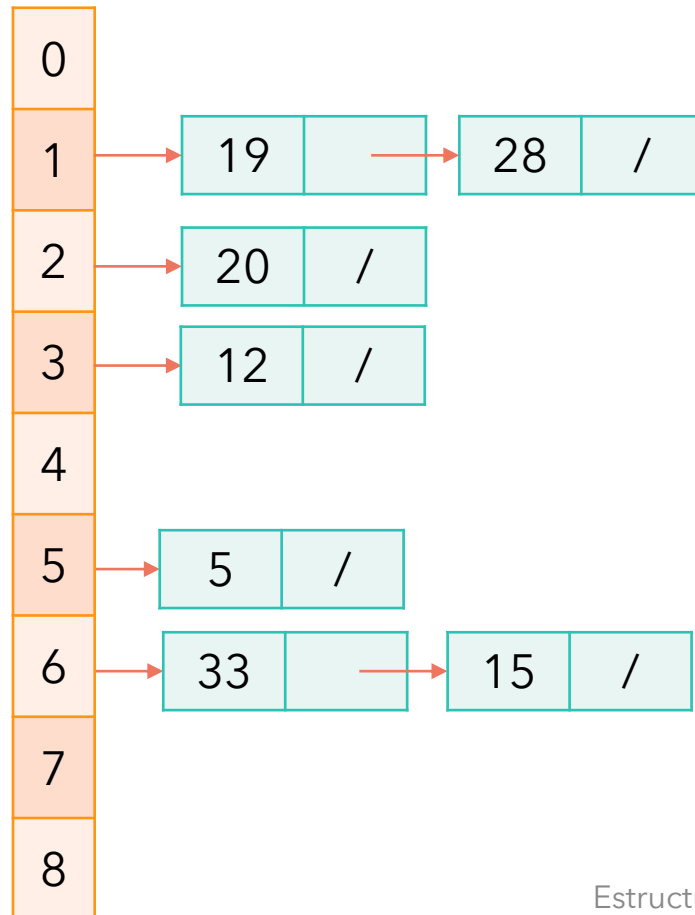


$$h(33) = 33 \bmod 9 = 6$$

# Tablas de dispersión

## Ejemplo

- Claves: 5, 28, 19, 15, 20, 33, 12, 17, 10



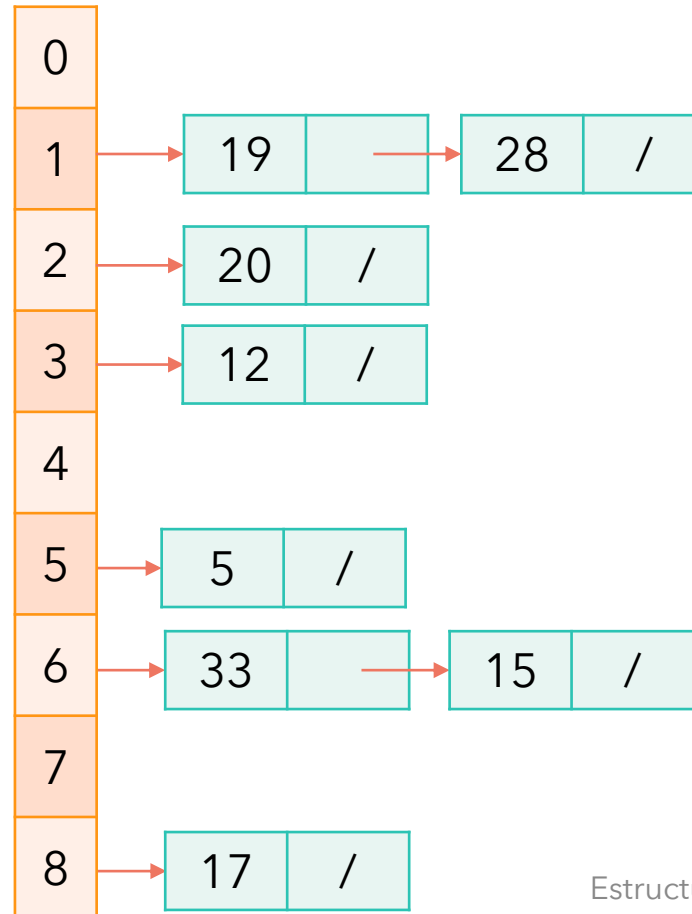
$$h(12) = 12 \bmod 9 = 3$$



# Tablas de dispersión

## Ejemplo

- Claves: 5, 28, 19, 15, 20, 33, 12, 17, 10

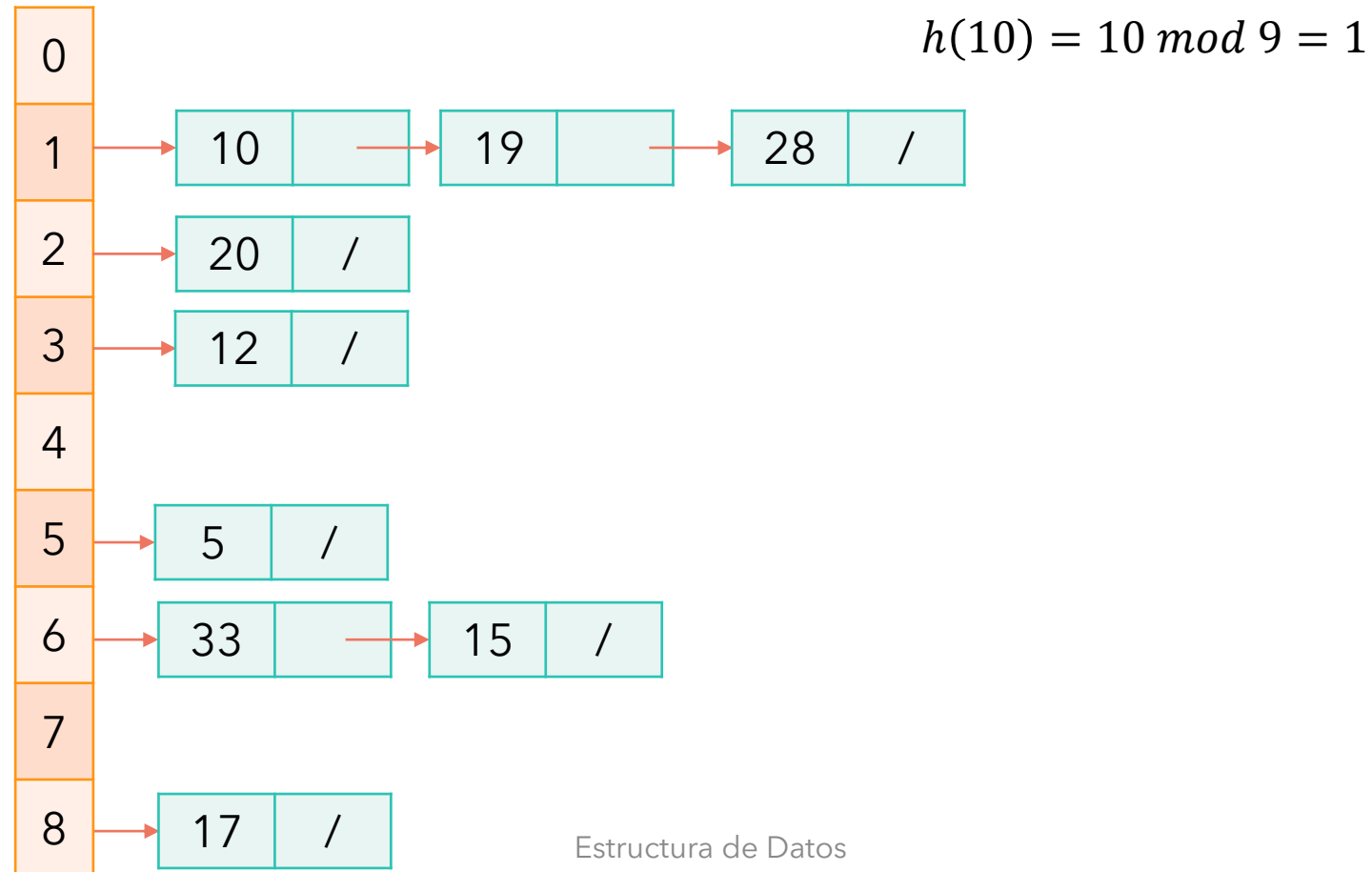


$$h(17) = 17 \bmod 9 = 8$$

# Tablas de dispersión

## Ejemplo

- Claves: 5, 28, 19, 15, 20, 33, 12, 17, 10

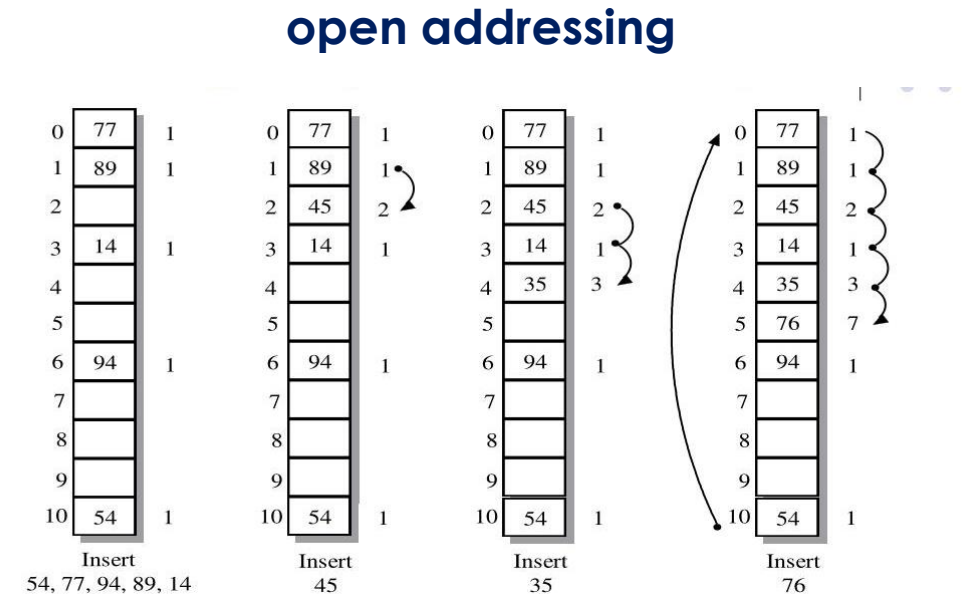
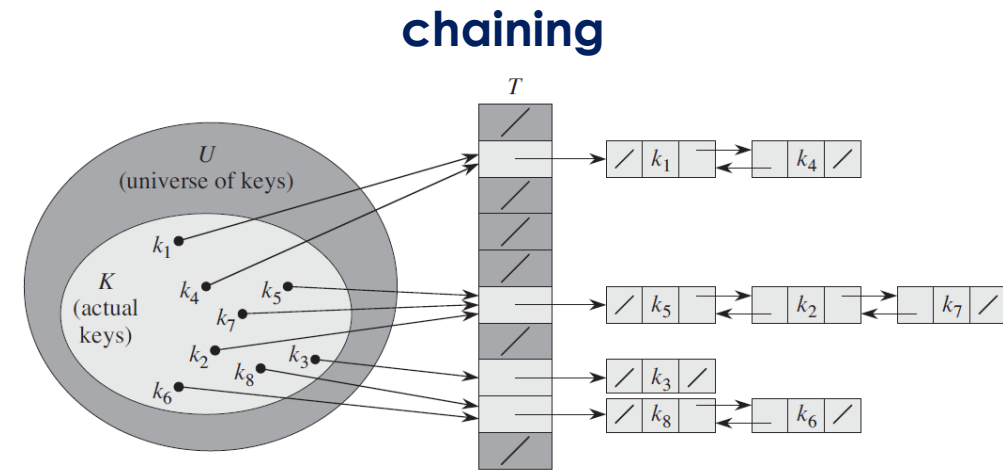


# Tablas de dispersión

❑ Esta asignación puede presentar problemas cuando dos claves se asignan a la misma posición en la tabla. Esta situación se denomina una **colisión**.

❑ Existen diferentes técnicas para solucionar colisiones en tablas de dispersión.

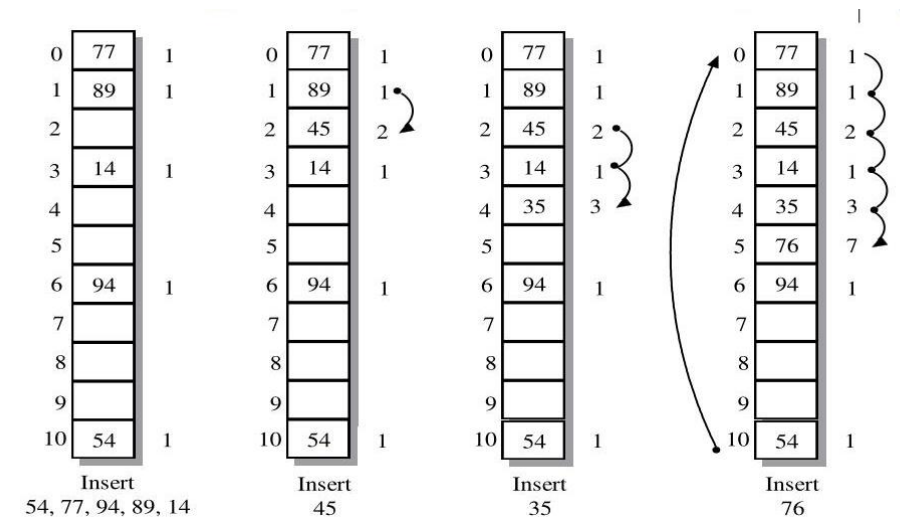
- Solución de colisiones por encadenamiento (**chaining**)
- Direccionamiento abierto (**open addressing**)



# Direccionamiento abierto

- ❑ Los elementos son almacenados dentro de la tabla, es decir, no se usa listas enlazadas.
- ❑ Cada posición de la tabla de dispersión contiene un elemento o NULL.
- ❑ Para insertar un elemento, se examina sucesivamente la tabla de dispersión hasta encontrar una posición vacía. Sin embargo, no se examinan todas las posiciones de la tabla (lo que tomaría  $O(n)$ ), solo las posiciones que dependen de la clave a ser insertada.

## open addressing



# Direcccionamiento abierto

- ❑ Con direccionamiento abierto, necesitamos para cada clave  $k$  una secuencia de prueba:

$$\langle h(k,0), h(k,1), \dots, h(k,m-1) \rangle$$

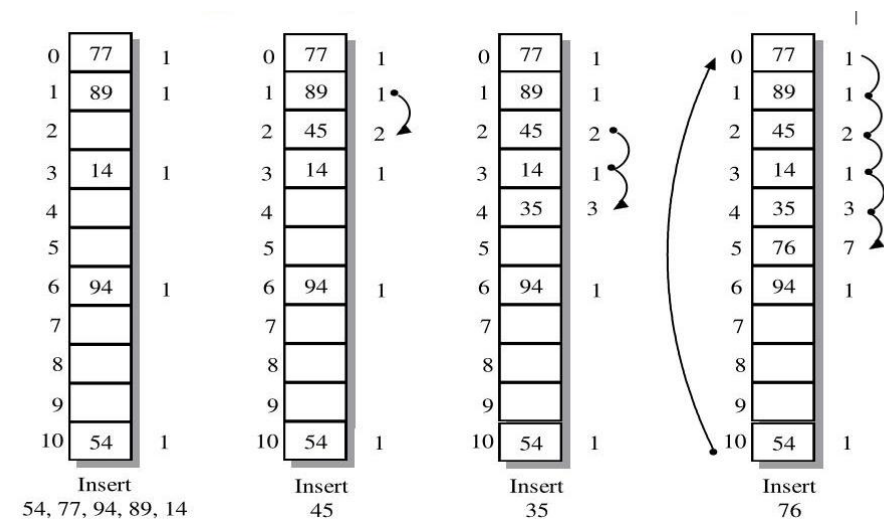
**HASH-INSERT( $T, k$ )**

```
1   $i = 0$ 
2  repeat
3       $j = h(k, i)$ 
4      if  $T[j] == \text{NIL}$ 
5           $T[j] = k$ 
6          return  $j$ 
7      else  $i = i + 1$ 
8  until  $i == m$ 
9  error "hash table overflow"
```

**HASH-SEARCH( $T, k$ )**

```
1   $i = 0$ 
2  repeat
3       $j = h(k, i)$ 
4      if  $T[j] == k$ 
5          return  $j$ 
6       $i = i + 1$ 
7  until  $T[j] == \text{NIL}$  or  $i == m$ 
8  return NIL
```

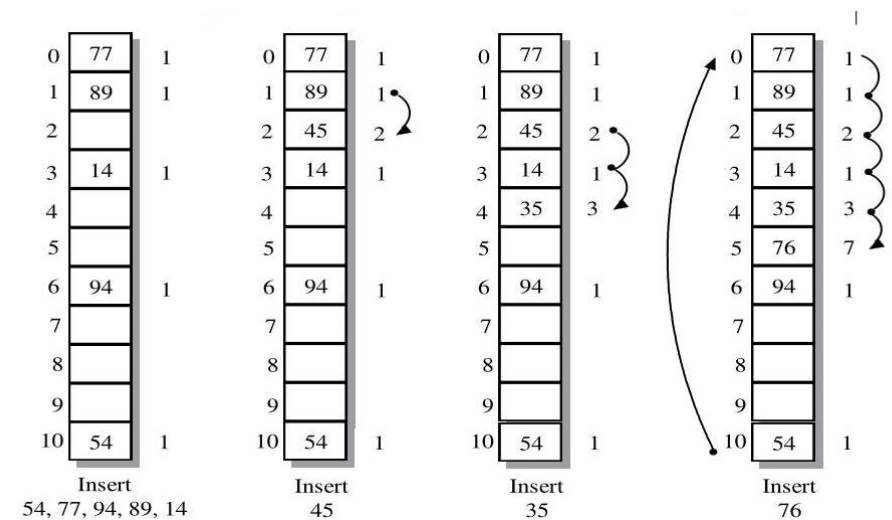
## open addressing



# Direccionamiento abierto

- ❑ Eliminar elementos de una tabla de dispersión con direccionamiento abierto es difícil.
- ❑ Ya que, si eliminamos un elemento de clave  $k$  en la posición  $i$ , y marcamos esta posición como NULL, no podremos encontrar otros elementos cuya operación de insertar encontraron la posición  $i$  ocupada.
- ❑ Usualmente, si se requiere eliminar elementos se usa la técnica de solución de colisiones por encadenamiento.

## open addressing



# Direccionamiento abierto

□ Generación de secuencia de prueba: se utiliza una función de dispersión como base

- Prueba lineal:

$$h(k,i) = (h'(k) + i) \bmod m$$

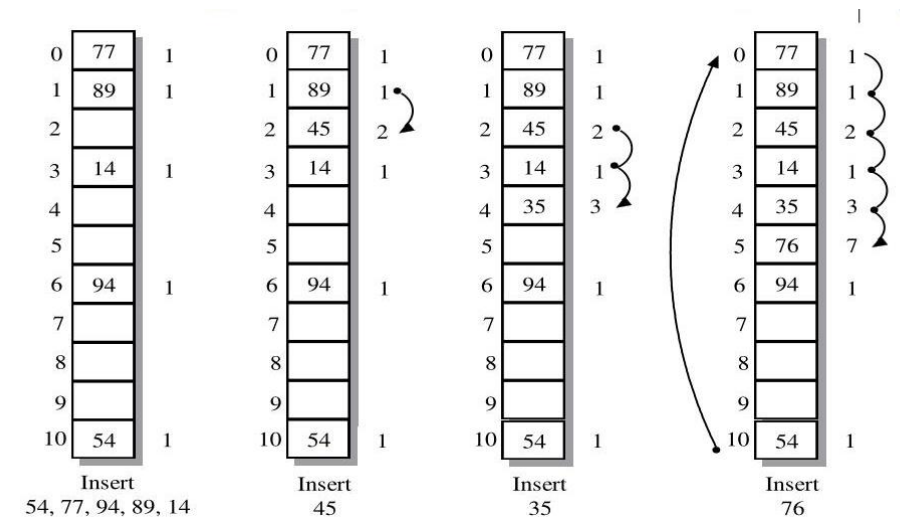
- Prueba cuadrática:

$$h(k,i) = (h'(k) + c_1 i + c_2 i^2) \bmod m$$

- Doble dispersión:

$$h(k,i) = (h_1(k) + i h_2(k)) \bmod m$$

## open addressing



# Direcccionamiento abierto

Ejemplo: Vamos a insertar las claves 10, 22, 31, 4, 15, 28, 17, 88, en una table hash con  $m=11$ , usando como función auxiliar

$$h'(k)=k$$

y la función de prueba cuadrática

$$h(k,i)=(h'(k)+c_1 i+c_2 i^2) \bmod m$$

con  $c_1=1$  y  $c_2=3$



# Direcccionamiento abierto

claves 10, 22, 31, 4, 15, 28, 17, 88

$$h'(k) = k$$

$$h(k, i) = (h'(k) + i + 3i^2) \bmod 11$$

0	
1	
2	
3	
4	
5	
6	
7	
8	
9	
10	

# Direcccionamiento abierto

claves 10, 22, 31, 4, 15, 28, 17, 88

$$h'(k) = k$$

$$h(k, i) = (h'(k) + i + 3i^2) \bmod 11$$

$$h(10, 0) = (h'(10) + 0 + 3 * 0^2) \bmod 11 = 10$$

0	
1	
2	
3	
4	
5	
6	
7	
8	
9	
10	10

# Direcccionamiento abierto

claves 10, 22, 31, 4, 15, 28, 17, 88

$$h'(k) = k$$

$$h(k, i) = (h'(k) + i + 3i^2) \bmod 11$$

$$h(22, 0) = (h'(22) + 0 + 3 * 0^2) \bmod 11 = 0$$

0	22
1	
2	
3	
4	
5	
6	
7	
8	
9	
10	10

# Direcccionamiento abierto

claves 10, 22, 31, 4, 15, 28, 17, 88

$$h'(k) = k$$

$$h(k, i) = (h'(k) + i + 3i^2) \bmod 11$$

$$h(31, 0) = (h'(31) + 0 + 3[0]^2) \bmod 11 = 9$$

0	22
1	
2	
3	
4	
5	
6	
7	
8	
9	31
10	10

# Direcccionamiento abierto

claves 10, 22, 31, 4, 15, 28, 17, 88

$$h'(k) = k$$

$$h(k, i) = (h'(k) + i + 3i^2) \bmod 11$$

$$h(4, 0) = (h'(4) + 0 + 3 * 0^2) \bmod 11 = 4$$

0	22
1	
2	
3	
4	4
5	
6	
7	
8	
9	31
10	10

# Direcccionamiento abierto

claves 10, 22, 31, 4, 15, 28, 17, 88

$$h'(k) = k$$

$$h(k, i) = (h'(k) + i + 3i^2) \bmod 11$$

$$h(15, 0) = (h'(15) + 0 + 3 * 0^2) \bmod 11 = 4$$

0	22
1	
2	
3	
4	4
5	
6	
7	
8	
9	31
10	10

# Direcccionamiento abierto

claves 10, 22, 31, 4, 15, 28, 17, 88

$$h'(k) = k$$

$$h(k, i) = (h'(k) + i + 3i^2) \bmod 11$$

$$\begin{aligned} h(15, 1) &= (h'(15) + 1 + 3 * 1^2) \bmod 11 \\ &= (15 + 1 + 3) \bmod 11 = 19 \bmod 11 = 8 \end{aligned}$$

0	22
1	
2	
3	
4	4
5	
6	
7	
8	15
9	31
10	10

# Direcccionamiento abierto

claves 10, 22, 31, 4, 15, 28, 17, 88

$$h'(k) = k$$

$$h(k, i) = (h'(k) + i + 3i^2) \bmod 11$$

$$h(28, 0) = (h'(28) + 0 + 3[0]^2) \bmod 11 = 6$$

0	22
1	
2	
3	
4	4
5	
6	28
7	
8	15
9	31
10	10



# Direcccionamiento abierto

claves 10, 22, 31, 4, 15, 28, 17, 88

$$h'(k) = k$$

$$h(k, i) = (h'(k) + i + 3i^2) \bmod 11$$

$$h(17, 0) = (h'(17) + 0 + 3 * 0^2) \bmod 11 = 6$$

0	22
1	
2	
3	
4	4
5	
6	28
7	
8	15
9	31
10	10

# Direcccionamiento abierto

claves 10, 22, 31, 4, 15, 28, 17, 88

$$h'(k) = k$$

$$h(k, i) = (h'(k) + i + 3i^2) \bmod 11$$

$$\begin{aligned} h(17, 1) &= (h'(17) + 1 + 3 * 1^2) \bmod 11 \\ &= (17 + 1 + 3) \bmod 11 = 20 \bmod 11 = 9 \end{aligned}$$

0	22
1	
2	
3	
4	4
5	
6	28
7	
8	15
9	31
10	10

# Direcccionamiento abierto

claves 10, 22, 31, 4, 15, 28, 17, 88

$$h'(k) = k$$

$$h(k, i) = (h'(k) + i + 3i^2) \bmod 11$$

$$\begin{aligned} h(17, 2) &= (h'(17) + 2 + 3 * 2^2) \bmod 11 \\ &= (17 + 2 + 12) \bmod 11 = 31 \bmod 11 = 9 \end{aligned}$$

0	22
1	
2	
3	
4	4
5	
6	28
7	
8	15
9	31
10	10

# Direcccionamiento abierto

claves 10, 22, 31, 4, 15, 28, 17, 88

$$h'(k) = k$$

$$h(k, i) = (h'(k) + i + 3i^2) \bmod 11$$

$$\begin{aligned} h(17, 3) &= (h'(17) + 3 + 3 * 3^2) \bmod 11 \\ &= (17 + 3 + 27) \bmod 11 = 47 \bmod 11 = 3 \end{aligned}$$

0	22
1	
2	
3	17
4	4
5	
6	28
7	
8	15
9	31
10	10

# Direcccionamiento abierto

claves 10, 22, 31, 4, 15, 28, 17, 88

$$h'(k) = k$$

$$h(k, i) = (h'(k) + i + 3i^2) \bmod 11$$

$$h(88, 0) = (h'(88) + 0 + 3 * 0^2) \bmod 11 = 0$$

0	22
1	
2	
3	17
4	4
5	
6	28
7	
8	15
9	31
10	10

# Direcccionamiento abierto

claves 10, 22, 31, 4, 15, 28, 17, 88

$$h'(k) = k$$

$$h(k, i) = (h'(k) + i + 3i^2) \bmod 11$$

$$\begin{aligned} h(88, 1) &= (h'(88) + 1 + 3 * 1^2) \bmod 11 \\ &= (88 + 1 + 3) \bmod 11 = 92 \bmod 11 = 4 \end{aligned}$$

0	22
1	
2	
3	17
4	4
5	
6	28
7	
8	15
9	31
10	10

# Direcccionamiento abierto

claves 10, 22, 31, 4, 15, 28, 17, 88

$$h'(k) = k$$

$$h(k, i) = (h'(k) + i + 3i^2) \bmod 11$$

$$\begin{aligned} h(88, 2) &= (h'(88) + 2 + 3 * 2^2) \bmod 11 \\ &= (88 + 2 + 12) \bmod 11 = 102 \bmod 11 = 3 \end{aligned}$$

0	22
1	
2	
3	17
4	4
5	
6	28
7	
8	15
9	31
10	10

# Direcccionamiento abierto

claves 10, 22, 31, 4, 15, 28, 17, 88

$$h'(k) = k$$

$$h(k, i) = (h'(k) + i + 3i^2) \bmod 11$$

$$\begin{aligned} h(88, 3) &= (h'(88) + 3 + 3 * 3^2) \bmod 11 \\ &= (88 + 3 + 27) \bmod 11 = 118 \bmod 11 = 8 \end{aligned}$$

0	22
1	
2	
3	17
4	4
5	
6	28
7	
8	15
9	31
10	10



# Direcccionamiento abierto

claves 10, 22, 31, 4, 15, 28, 17, 88

$$h'(k) = k$$

$$h(k, i) = (h'(k) + i + 3i^2) \bmod 11$$

$$\begin{aligned} h(88, 4) &= (h'(88) + 4 + 3 * 4^2) \bmod 11 \\ &= (88 + 4 + 48) \bmod 11 = 140 \bmod 11 = 8 \end{aligned}$$

0	22
1	
2	
3	17
4	4
5	
6	28
7	
8	15
9	31
10	10

# Direcccionamiento abierto

claves 10, 22, 31, 4, 15, 28, 17, 88

$$h'(k) = k$$

$$h(k, i) = (h'(k) + i + 3i^2) \bmod 11$$

$$\begin{aligned} h(88, 5) &= (h'(88) + 5 + 3 * 5^2) \bmod 11 \\ &= (88 + 5 + 75) \bmod 11 = 168 \bmod 11 = 3 \end{aligned}$$

0	22
1	
2	
3	17
4	4
5	
6	28
7	
8	15
9	31
10	10

# Direcccionamiento abierto

claves 10, 22, 31, 4, 15, 28, 17, 88

$$h'(k) = k$$

$$h(k, i) = (h'(k) + i + 3i^2) \bmod 11$$

$$\begin{aligned} h(88, 6) &= (h'(88) + 6 + 3 * 6^2) \bmod 11 \\ &= (88 + 6 + 108) \bmod 11 = 202 \bmod 11 = 4 \end{aligned}$$

0	22
1	
2	
3	17
4	4
5	
6	28
7	
8	15
9	31
10	10

# Direcccionamiento abierto

claves 10, 22, 31, 4, 15, 28, 17, 88

$$h'(k) = k$$

$$h(k, i) = (h'(k) + i + 3i^2) \bmod 11$$

$$\begin{aligned} h(88, 7) &= (h'(88) + 7 + 3 * 7^2) \bmod 11 \\ &= (88 + 7 + 147) \bmod 11 = 242 \bmod 11 = 0 \end{aligned}$$

0	22
1	
2	
3	17
4	4
5	
6	28
7	
8	15
9	31
10	10

# Direcccionamiento abierto

claves 10, 22, 31, 4, 15, 28, 17, 88

$$h'(k) = k$$

$$h(k, i) = (h'(k) + i + 3i^2) \bmod 11$$

$$\begin{aligned} h(88, 8) &= (h'(88) + 8 + 3 * 8^2) \bmod 11 \\ &= (88 + 8 + 192) \bmod 11 = 288 \bmod 11 = 2 \end{aligned}$$

0	22
1	
2	88
3	17
4	4
5	
6	28
7	
8	15
9	31
10	10

# Tablas de dispersión

## Aplicaciones

- ❑ Criptografía: resumen de mensajes
- ❑ Archivos de sistema: enlaza el nombre con el path
- ❑ Verificación de contraseñas
- ❑ Búsqueda de patrones en cadenas
- ❑ Python: diccionarios
- ❑ Compiladores: palabras clave

