



# Estructura de Datos Avanzadas

Conceptos y actividades correspondientes al primer corte.

# Unidades de aprendizaje

## 1. Búsqueda

- Métodos de búsqueda.
- Métodos de ordenamiento y su aplicación.
- Cálculo de la complejidad de la búsqueda secuencial y la búsqueda binaria.

# Transformación de claves

## Concepto:

Permite aumentar la velocidad de búsqueda sin necesidad de tener los elementos ordenados. Cuenta con la ventaja de que el tiempo de búsqueda es prácticamente independiente del número de componentes del arreglo.

Para los métodos descritos anteriormente su complejidad y el tiempo de búsqueda es proporcional al número de componentes del arreglo. Es decir, a mayor número de elementos se realiza un mayor número de comparaciones.

## Conjunto de datos

103027	Barrientos Rosas Pedro
103282	Cruz López Giovanni
103164	Díaz Beltrán Pedro

Clave	Dato
-------	------

Sería ideal poder tener acceso a los datos de manera directa, es decir, sin tener que recorrer algunos elementos antes de llegar al elemento buscado.

# Transformación de claves

## Función de transformación:

Este método permite acceder a los elementos de un conjunto de datos de manera directa ya que trabaja basándose en una función de transformación conocida como **función hash (H)** que convierte una clave dada en una dirección (índice) dentro del arreglo.

### Caso de aplicación

Se necesita almacenar la información relacionada a 100 alumnos cuyas matrículas son del 1 al 100. Para esto se define un arreglo de 100 elementos con índices numéricos comprendidos entre los valores 1 y 100.

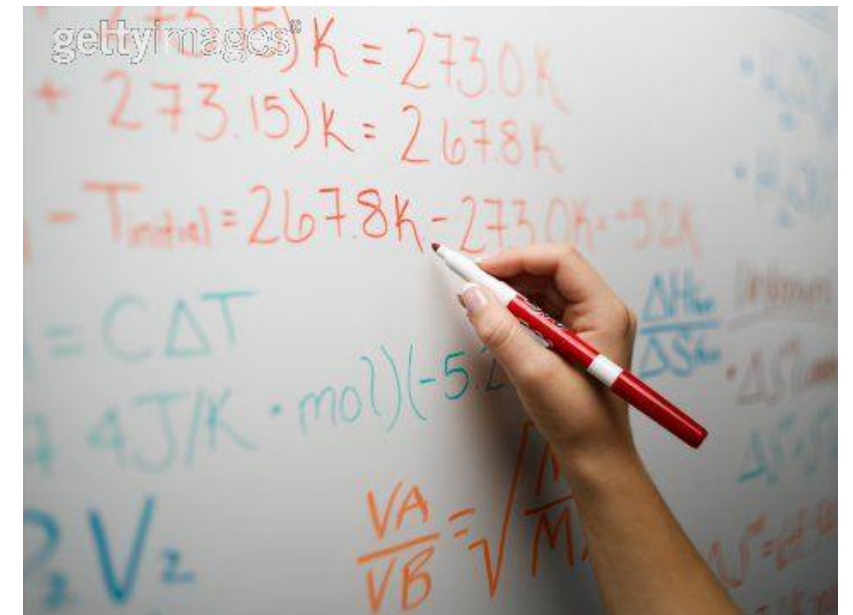
Se necesita almacenar la información relacionada a 100 empleados cuya clave será su número de seguro social que esta formado por once dígitos.

Es ineficiente definir un arreglo con 99 999 999 999 elementos para almacenar solamente 100 datos y el costo de memoria es excesivo.

# Transformación de claves

## Objetivos de la transformación de claves

- Consiste en la aplicación de una función de transformación o función hash a los valores que se van a insertar, para saber en que posición del arreglo quedarán
- Un método para resolver colisiones, el hecho de que dos valores distintos sean asignados a la misma dirección del arreglo por causa de la función hash, hecho que se conoce como colisión.



## Selección de la función hash

Es uno de los procesos más importantes, sin embargo, no hay reglas que permitan determinar cuál será la función más apropiada para un conjunto de claves, de tal manera que asegure la reducción de colisiones.

- Función Módulo
- Función Cuadrado
- Función Truncamiento
- Función Plegamiento

# Transformación de claves

## Función Módulo o División

Consiste en tomar el residuo de la división de la clave entre el número de componentes del arreglo. Suponga que se tiene un arreglo de  $N$  elementos y  $K$  es la clave del dato a buscar.

La función hash queda:

$$H(k) = (K \bmod N) + 1$$

Para lograr una mayor uniformidad en la distribución,  $N$  debe ser un número primo. (El número primo próximo a  $N$ )

### Ejemplo:

Sea  $N=100$ , el tamaño del arreglo  
Sus direcciones de 1-100.

Sea  $K1 = 7259$

$K2 = 9359$

Dos claves que deban asignarse posiciones en el arreglo

$$H(K1) = (7259 \bmod 100) + 1 = 60$$

$$H(K2) = (9359 \bmod 100) + 1 = 60$$

Donde  $H(K1)$  es igual a  $H(K2)$  y  $K1$  es distinto de  $K2$ , es una colisión



# Transformación de claves

## Función Cuadrado

Consiste en elevar al cuadrado la clave y tomar los dígitos centrales como dirección.

El número de dígitos a tomar queda determinado por el rango del índice.

La función hash queda:

$$H(k) = \text{dígitos centrales } (K^2) + 1$$

Para lograr una mayor uniformidad en la distribución, N debe ser un número primo. (El número primo próximo a N)

### Ejemplo:

Sea  $N=100$ , el tamaño del arreglo  
Sus direcciones de 1-100.

Sea  $K_1 = 7259$

$K_2 = 9359$

Dos claves que deban asignarse posiciones en el arreglo

$$H(K_1^2) = \text{dígitos\_centrales}(52693081) + 1 = 94$$

$$H(K_2^2) = \text{dígitos\_centrales}(87590881) + 1 = 91$$

# Transformación de claves

## Función Plegamiento

Consiste en dividir la clave en partes de igual número de dígitos (La última puede tener menos dígitos) y operar con ellos tomando como dirección los dígitos menos significativos.

La operación entre las partes puede hacerse por medio de sumas o multiplicaciones.

$$H(k) = \text{digitos\_menos\_significativos}((d1..di) + (di+1..dj) + \dots + (d1..dn)) + 1$$

### Ejemplo:

Sea  $N=100$ , el tamaño del arreglo

Sus direcciones de 1-100.

Sea  $K_1 = 7259$

$K_2 = 9359$

Dos claves que deban asignarse posiciones en el arreglo

Se toman solamente dos dígitos porque el arreglo tiene cien elementos.

$$H(K_1) = \text{DígitosMenosSignificativos}(72+59)+1 = (131)+1 = \mathbf{32}$$

$$H(K_2) = \text{DígitosMenosSignificativos}(93+59)+1 = (152)+1 = \mathbf{53}$$



# Transformación de claves

## Función Truncamiento

Consiste en tomar algunos de la clave y formar con ellos una dirección.

Este método es de los más sencillos, pero es también de los que ofrece menos uniformidad en la distribución de las claves.

Se pueden elegir los dígitos de las posiciones pares o impares.  $H(k) = \text{elegir\_digitos}(d1, d2, \dots, d_n) + 1$

### Ejemplo:

Sea  $N=100$ , el tamaño del arreglo  
Sus direcciones de 1-100.

Sea  $K1 = 7\ 259$

$K2 = 9\ 359$

Dos claves que deban asignarse posiciones en el arreglo

$$H(K_1) = \text{Elegir\_dígitos}(7259) + 1 = 76$$

$$H(K_2) = \text{Elegir\_dígitos}(9359) + 1 = 96$$

Se toman el primer y tercer número de la clave y se une de izquierda a derecha

# Transformación de claves

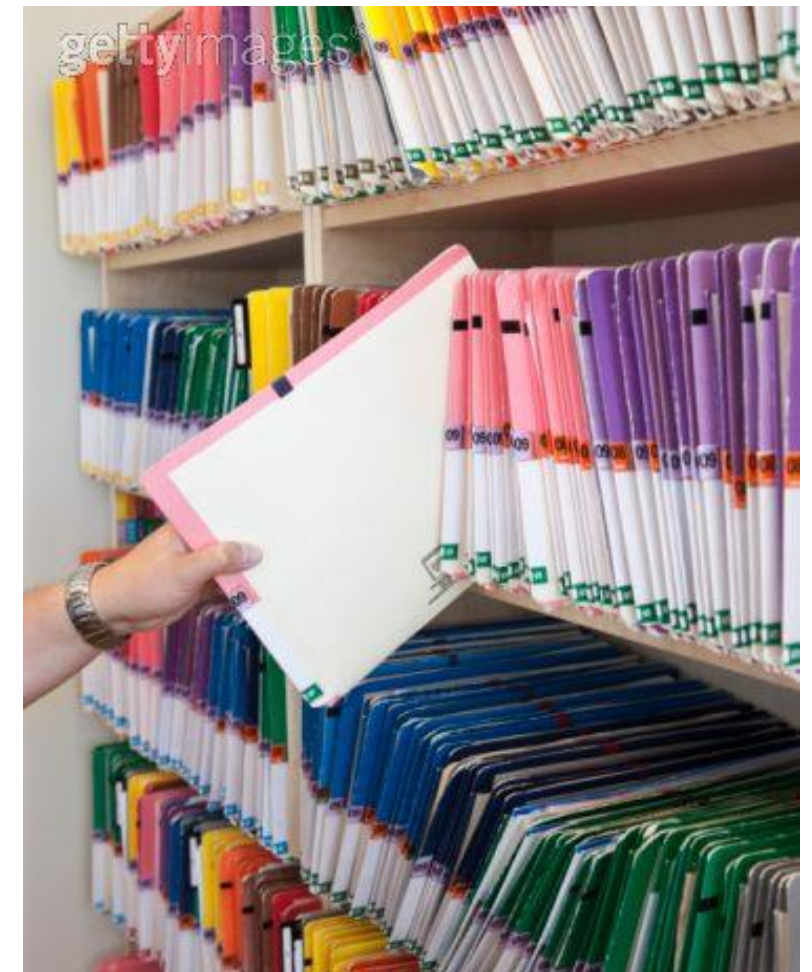
## Solución de Colisiones

- Reasignación
- Arreglos Anidados
- Encadenamiento

### Reasignación

Propone varios enfoques que trabajan bajo el esquema de comparación y reasignación de elementos.

- Prueba lineal
- Prueba cuadrática
- Doble dirección hash



# Transformación de claves

## Reasignación

**Prueba Lineal:** Consiste en recorrer el arreglo secuencialmente a partir del punto de colisión buscando el elemento. Concluye cuando el elemento es hallado o encuentra una posición vacía. El arreglo es circular.

### Algoritmo:

```
{Este algoritmo busca el dato con clave K en el arreglo V de N elementos. Resuelve el problema de las
colisiones por medio de la prueba lineal}
{ D y DX son variables de tipo entero}
4. D=H[K] {Genera dirección}
5. Si (V[D] == K) entonces
    Escribir "El elemento esta en la posición D"
    Si no
        Hacer DX=D+1;
    2.1 Repetir mientras (DX <= N) y (V[DX]<> K) y (V[DX]<>VACIO) y (DX<>D)
        Hacer DX=DX+1;
        2.1.1 Si (DX==N+1)
            Entonces Hacer DX=1
            Si no Hacer DX=Cen-1
        2.1.2 {Fin del paso 2.1.1}
    2.2{fin del ciclo del paso 2.1}
    2.3 Si (V[DX] == K) entonces
        Escribir "El elemento esta en la posición DX"
        Si no
            Escribir "El elemento no está en el arreglo"
    2.4{ Fin del condicional del paso 2.3}
3 {Fin del condicional del paso 2}
```

# Transformación de claves

**Ejemplo:** Sea V un arreglo de diez elementos con las siguientes claves:

**V = [ 25, 43, 56, 35, 54, 13, 80, 104]**       $H(K) = (K \bmod 10) + 1$

1	80
2	
3	
4	43
5	54
6	25
7	56
8	35
9	13
10	104

K	H(K)
25	6
43	4
56	7
35	6
54	5
13	4
80	1
104	5

D	DX
6	7 8

**K=35**

D	DX
4	5 6 7 8 9

**K=13**



# Transformación de claves

## Reasignación

**Prueba Cuadrática:** Es similar al enfoque de prueba lineal pero con saltos cuadráticos de las direcciones que se generan.

- $D+(1)^2, D+(2)^2, D+(3)^2, +\dots D+i^2$
- $D+1, D+4, D+9, \dots D+i^2$

## Algoritmo:

```
{Este algoritmo busca el dato con clave K en el arreglo V de N elementos. Resuelve el problema de las
colisiones por medio de la prueba cuadrática}
{ D, DX e I son variables de tipo entero}
4. D=H[K] {Genera dirección}
5. Si (V[D] == K) entonces
    Escribir "El elemento esta en la posición D"
    Si no
        Hacer I = 1 y DX = D + I2;
    2.1 Repetir mientras (V[DX] <> K) y (V[DX] <> VACÍO)
        Hacer I = I + 1; DX = D + I2;
        2.1.1 Si (DX > N)
            Entonces Hacer I=0; DX=1 y D=1;
        2.1.2 {Fin del paso 2.1.1}
    2.2 {fin del ciclo del paso 2.1}
    2.3 Si (V[DX] == K) entonces
        Escribir "El elemento esta en la posición DX"
        Si no
            Escribir "El elemento no está en el arreglo"
    2.4 { Fin del condicional del paso 2.3}
3 {Fin del condicional del paso 2}
```

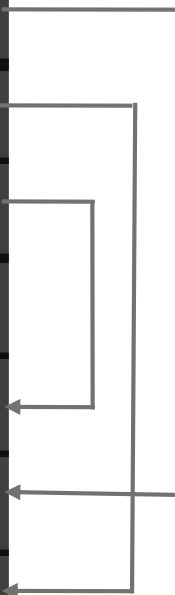


# Transformación de claves

**Ejemplo:** Sea V un arreglo de diez elementos con las siguientes claves:

**V = [ 25, 43, 56, 35, 54, 13, 80, 104, 55]**       $H(K) = (K \bmod 10) + 1$

1	80
2	55
3	
4	43
5	54
6	25
7	56
8	13
9	104
10	35



K	H(K)
25	6
43	4
56	7
35	6
54	5
13	4
80	1
104	5
55	6

D	I	DX
6	1	7
	2	10
	3	15
1	0	1
	1	2

**K=55**

# Transformación de claves

## Reasignación

**Doble Dirección Hash:** Consiste en generar otra dirección aplicando la misma función hash a la dirección previamente obtenida. El proceso se detiene cuando el elemento es hallado o se encuentra una posición vacía.

### Algoritmo:

#### **DobleDireccion(V, N, K)**

{Este algoritmo busca el dato con clave K en el arreglo V de N elementos. Resuelve el problema de las colisiones por medio de la doble dirección hash}

{ D y DX son variables de tipo entero}

4.  $D = H[K]$  {Genera dirección}

5. Si  $(V[D] == K)$  entonces

    Escribir "El elemento esta en la posición D"

Si no

    Hacer  $DX = H'(D)$ ;

2.1 Repetir mientras  $(DX \leq N)$  y  $(V[DX] \neq K)$  y  $(V[DX] \neq \text{VACÍO})$  y  $(DX \neq D)$

        Hacer  $DX = H'(DX)$ ;

2.2 {fin del ciclo del paso 2.1}

2.3 Si  $(V[DX] == K)$  entonces

    Escribir "El elemento esta en la posición DX"

Si no

    Escribir "El elemento no está en el arreglo"

2.4 { Fin del condicional del paso 2.3}

3 {Fin del condicional del paso 2}