



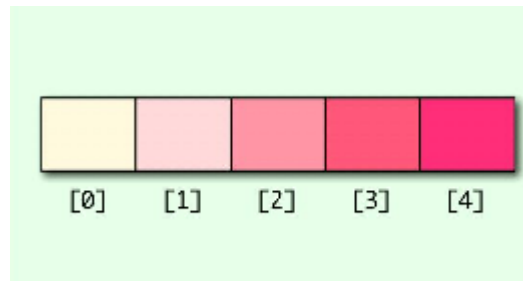
UNIVERSIDAD INCA GARCILASO DE LA VEGA

Facultad de Ingeniería de Sistemas, Computo y Telecomunicaciones

Asignatura
Estructura de Información

Tema

Arreglos



Profesor: Carlos A. Ruiz De La Cruz Melo

Correo: ruizdelacruzmelo@uigv.edu.pe

Definición

es una estructura de datos que almacena bajo el mismo nombre (variable) a una colección de datos del mismo tipo.

Características

Almacenan los elementos en posiciones contiguas de memoria

Tienen un mismo nombre de variable que representa a todos los elementos.

Uso de índice.

Tipos de arreglos

Unidimensionales / vectores

vec

10	5	20	80
----	---	----	----

Bidimensionales / matrices

mat

56	7
9	10

Multidimensionales

mul

8	15	12
20	5	2

Información del arreglo

**vector de
enteros**



10	5	20	80
----	---	----	----

**vector de
cadenas**



FISC	UNMSM	UNI	USIL
------	-------	-----	------

**vector de
reales**



3.4	70.0	20	4.4
-----	------	----	-----

**vector de
TAD**



09087 Carlos 10	05678 María 15	02013 Ana 11	08077 Pablo 17
-----------------------	----------------------	--------------------	----------------------

Características de un arreglo

Para un vector de enteros

FINITO

```
int DATO[ 4 ] ;
```

DATO

HOMOGENEO

Datos del mismo tipo

10 5 20 80

0 1 2 3

ORDENADO

Uso de índices

Operaciones

Lectura / escritura

Ordenación

Inserción

Eliminación

Modificación

Recorrido

Búsqueda

Primitivas

PRIMITIVA	para el vector T con N elementos y una variable VALOR
inicio	$i \leftarrow 0$
lectura	$VALOR \leftarrow T[i]$
avanzar	$i \leftarrow i + 1$
Verificar final	$i < N$
Escribir	$T[i] \leftarrow VALOR$
terminar	NO EXISTE

Ventajas

Costo de acceso a un elemento es constante

Estructura de fácil uso

Desventajas

Tamaño constante

Búsqueda lenta en arreglo desordenado

Insertar o eliminar toma mucho tiempo

Vector de TAD

PRIMERO :

definir el TAD

especificación ALUMNO

variable

entero : codigo

cadena : nombre

operaciones

IngresarAlumno : no retorna valor

MostrarAlumno : no retorna valor

Registrar(a, N, x) : no retorna valor

Visualizar(a , N) : no retorna valor

EliminarPorPosicion(a , N, p) : retorna un booleano

significado

EliminarPorPosicion elimina del vector **a** de una cantidad **N** de elementos un alumno en una posicion **p**.


:

Fin_ALUMNO

Vector de TAD

SEGUNDO :

construir el algoritmo



```
funcion EliminarPorPosicion(a , N, p): logico
  Eliminar ← falso
  i ← 0
  mientras (i < N) y ( no Eliminar) hacer
    si (i = p) entonces
      Eliminar ← verdadero
    sino
      i ← i + 1
  fin si
  fin mientras
  si (Eliminar = verdadero) entonces
    mientras ( i < (N-1)) hacer
      a[i] ← a[i+1]
      i ← i + 1
    fin mientras
    N ← N - 1
  fin si
  retornar Eliminar
fin EliminarPorPosicion
```

Se define un vector **a** de tamaño 6

ENTRADA:

a	08970	05122	02440	08120		
	carlos	maría	elena	juan		
	0	1	2	3	4	5

N= 4

p= 1

Cantidad de elementos
que tiene el vector en
un instante

Elemento en una
posición para eliminar

PROCESO:

a

08970
carlos

05122
maría

02440
elena

08120
juan

0

1

2

3

4

5

p= 1

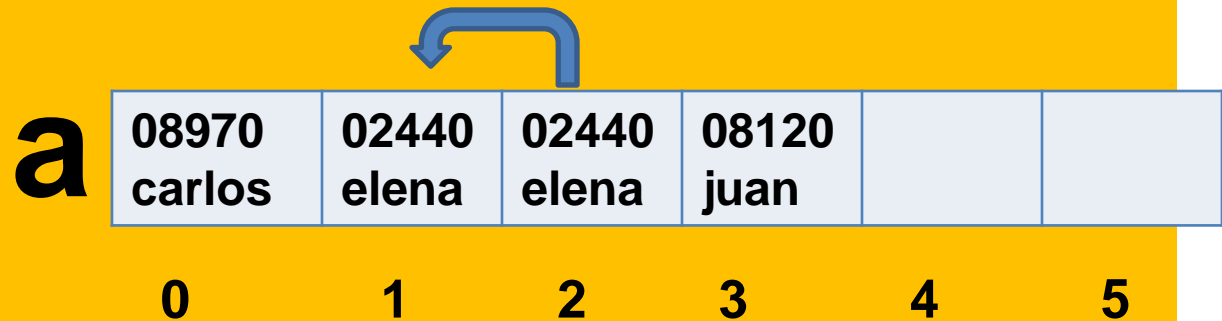
i= 1



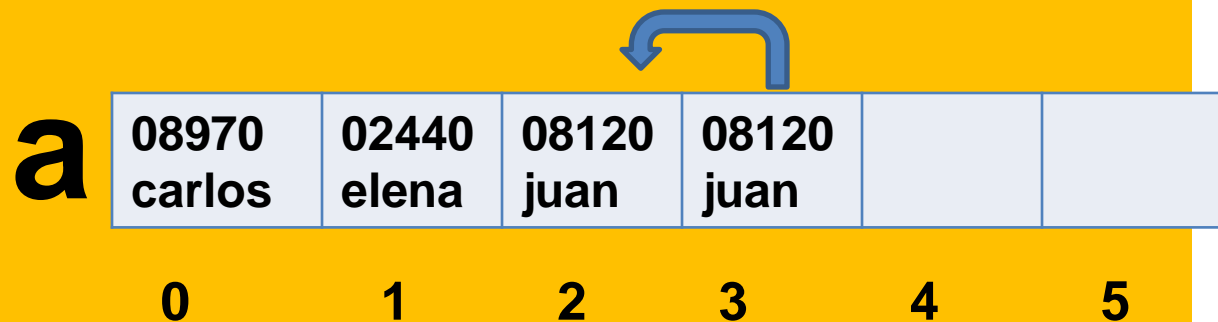
**Elemento a
eliminar**

PROCESO:

Se copia el elemento en posición 2 en la posición 1



Se copia el elemento en posición 3 en la posición 2



Finalmente $N = 3$, para reconocer solo 3 elementos

DISPERSION EN UN ARREGLO

Todos los elementos identificados por una clave que debe de ser única.

El método por transformación de claves accesa a los datos por una dirección (índice).

Está basado en una función de transformación, cálculo de dirección o función hash.

$$\textit{Dirección} = H(\textit{clave})$$

❑ Convierte una clave en una dirección que permite acceder **directamente** a su contenido.

INTRODUCCIÓN

La clave que es tomada para calcular debe ser mayor o igual que el número de elementos del arreglo.

Debe ser simple de calcular y debe asignar direcciones de manera uniforme.

Si $H(K_1)=D$, $H(K_2)=D$ y $K_1 \neq K_2$, entonces se dice que hay una **colisión**.

Una **colisión** es la asignación de una misma dirección a dos o más claves distintas.

FUNCIONES DE CÁLCULO DE DIRECCIÓN

❑ Para seleccionar una función hash:

- Importante definirla y difícil a la vez.
- Difícil de determinar.
- No existen reglas.

❑ Funciones:

- Módulo (por División).
- Cuadrado.
- Plegamiento o Doblamiento.
- Truncamiento.

FUNCIONES DE MÓDULO (POR DIVISIÓN)

Se toma el resto de la división de la clave entre el número de componentes del arreglo.

Al residuo de la división se le suma 1, para obtener un valor entre 1 y N.

Si tenemos un arreglo de N elementos, y la clave a buscar es K, la función estará definida así:

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

Para lograr una mayor uniformidad en la distribución, N debe ser primo o divisible entre pocos números.

FUNCIONES DE MÓDULO (POR DIVISIÓN)

ejemplo

Sean $N = 100$ el tamaño del arreglo, y sean sus direcciones los números entre 1 y 100.

Sean $K_1 = 7259$ y $K_2 = 9359$ dos claves a las que se debe asignar posiciones en el arreglo.

Aplicando la fórmula con $N = 100$:

- $H(K_1) = (7259 \bmod 100) + 1 = 60$
- $H(K_2) = (9359 \bmod 100) + 1 = 60$

En este caso hay una colisión.

FUNCIONES DE MÓDULO (POR DIVISIÓN)

Se aplica la fórmula con N igual a un valor primo:

- $H(K_1) = (7259 \bmod 97) + 1 = 82$
- $H(K_2) = (9359 \bmod 97) + 1 = 48$

Con $N = 97$ se ha eliminado la colisión.

RESOLUCIÓN DE COLISIONES

La elección del método adecuado es importante.

Método de reservar una casilla por clave.

Métodos más usados:

- Reasignación.
- Arreglos anidados.
- Encadenamiento.

Resolución de Colisión por Reasignación

- Prueba Lineal.
- Prueba Cuadrática.
- Doble Dirección hash.

REASIGNACIÓN POR PRUEBA LINEAL

Detectada la colisión, se recorre el arreglo secuencialmente a partir del punto de colisión, buscando el elemento.

El proceso concluye cuando se halla el elemento o se llega a una posición vacía.

Se trata el arreglo como una estructura circular. El elemento que sigue al último es el primero.

Existe fuerte agrupamiento alrededor de ciertas claves.

A concentraciones de claves muy frecuentes, la búsqueda tenderá a ser secuencial.

REASIGNACIÓN POR PRUEBA LINEAL

V

1	
2	
3	
4	43
5	
6	25
7	56
8	
9	
10	


K	H(K)
25	6
43	4
56	7
35	6

Ya esta ocupado
por otra clave

REASIGNACIÓN POR PRUEBA LINEAL

V

1	
2	
3	
4	43
5	
6	25
7	56
8	35
9	
10	



K	H(K)
25	6
43	4
56	7
35	6

REASIGNACIÓN POR PRUEBA LINEAL

V

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


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

Ya esta ocupado
por otra clave

REASIGNACIÓN POR PRUEBA LINEAL

V

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



A diagram consisting of a vertical line with a horizontal segment at the top and bottom. The top horizontal segment is connected to the right side of the cell at index 4, and the bottom horizontal segment is connected to the left side of the cell at index 9. An arrow points from the top horizontal segment down to the bottom horizontal segment, indicating a linear search path.

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

REASIGNACIÓN POR PRUEBA LINEAL

V

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

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

Ya esta ocupado
por otra clave

REASIGNACIÓN POR PRUEBA LINEAL

V

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

REASIGNACIÓN POR PRUEBA LINEAL

Algoritmo PruebaLineal(K)

$D \leftarrow H(K)$

Si $(v(D) = K)$ **entonces**

Escribir “El elemento está en la posición D”

Sino

Si $(v(D) = \text{Vacio})$ **entonces**

$v(D) \leftarrow K$

Sino

$DX \leftarrow D + 1$

Mientras $((v(DX) \neq K) \text{ y } (v(DX) \neq \text{Vacío}) \text{ y } (DX \neq D))$ **hacer**

$DX \leftarrow DX + 1$

Si $(DX = nElementos + 1)$ **entonces**

$DX \leftarrow 1$

Finsi

finmientras

Si $(v(DX) = K)$ **entonces**

Escribir “El elemento está en la posición DX”

Sino

Si $(DX = D)$ **entonces**

Escribir “El arreglo no tiene casillas vacías”

Sino

$v(D) \leftarrow K$

finsi

finsi

finsi

finsi

finPruebaLineal