Algorítmica III



Técnicas de dispersión

INTEGRANTES:

BELTRÁN NUÑOVERO, Anderson 17200062

ESQUIVEL GRADOS, Luis Germán 17200154

PROFESOR: CORTEZ VASQUEZ, Augusto

CICLO: 2019-I

FACULTAD: Ingeniería de Sistemas e Informática (FISI)

AÑO:

**2019**

**Técnicas de dispersión**

**Dispersión Cerrada**

V, n, elem

s

Entrada: Vector V long n Tipo entero

n: entero (tamaño del vector y módulo)

elem: entero (elemento a insertar)

Salida: Cadena s

Precondición: n>0

Función: insertar(V:vector) DEV(s:Cadena)

Postcondición:

Si Ɐi ∈ {0,…,n-1}: V[i] != elem ∧ ∃ j ∈ {0,…,n-1} / (elem%n=j) ∧ V[j]=-1→ (V[j] ←elem ∧ s ← "Insertado correctamente en el índice "+ j)

Sino, si: Ɐi ∈ {0,…,n-1}: V[i] != elem ∧ ∃ j ∈ {0,…,n-1} / (elem%n=j) ∧ ∃ k ∈ {j+1,…,n-1} / Ɐm ∈ {j+1,…,k-1}: V[m] != -1 ∧ V[k]= -1 -> (V[k] ← elem ∧ s ← "Insertado correctamente en el índice "+ k)

Sino, si: Ɐi ∈ {0,…,n-1}: V[i] != -1 -> s ←”Vector lleno”

Sino, si: ∃i ∈ {0,…,n-1} / V[i]=elem → s ←”ya existe”

Nota:

Cada elemento del vector está **inicializado en -1**, esto significa que **no tiene valor asignado** o **es un espacio vacío**.

Accion insertar(V)

Inicio

Función Insertar:

Depende de la función BPI(búsqueda para insertar)

Si BPI!=-1 o -2 entonces se adicionará el valor en el índice correspondiente; caso contrario mostrará el error ocurrido.

indice ←BPI(elem)

Si (indice ←-1)

s ←”ya existe”

Sino

Si (indice=-2)

s ←”Vector lleno”

Sino

V[indice] ←elem

s ←”Insertado correctamente”

Fin si

Fin si

Fin Accion

Función funDisp:

Es la función de dispersión.

En el algoritmo implementado esta función es **módulo n**. Donde **n=10.**

Accion funcDisp(elem)

Inicio

Retorna elem%n

Fin Accion

Funcion BPI(elem)

Función BPI(búsqueda para insertar):

Este algoritmo busca la posición para insertar el valor, respetando la función de dispersión.

Inicio

indice ←-1

i ←funcDisp(elem)

Si (i ←0)

Se usa la variable **lim** para especificar el limite **hasta donde llegara la búsqueda**.

En caso i sea cero, se le asignará un límite en el que no tenga que “dar la vuelta al vector”.

lim ←n-1

Sino

lim ←i-1

Fin Si

Mientras (i!=lim)

Si, **el elemento ya existe**, entonces se **retornará -1**.

Si (V[i] ←elem)

Retornar -1

Sino

Caso contrario, se evaluara si **existe un espacio vacio** y si el **indice sigue siendo -1**(ningún espacio vacio anterior al actual); si ese es el caso el **indice pasara a tomar el valor de i**.

Si (V[i] ←-1)

Si (indice ←-1)

indice ←i

Fin Si

Fin Si

Condicional para que se siga evaluando los siguientes espacios, si i llego al final del vector; seguirá buscando en el inicio del vector.

Si (i ←n-1)

i ←0

Sino

i++

Fin Si

Fin Si

Fin Si

Fin Mientras

Si (indice ←-1)

Condicional si indice sigue siendo -1 eso quiere decir que **no encontró ningún espacio vacio**, por lo tanto le asignamos un nuevo valor (**-2**), para indicar que **el vector esta lleno**.

Retornar -2

Sino

Retorna indice

Fin Accion