Vectores o Dimensiones 2 . Actualizacion

CFGS: ASIR. Implantación de Sistemas Operativos

Arreglos unidimensionales: Operaciones con Vectores

Las operaciones que se pueden realizar con vectores durante el proceso de resolución de un problema usando la programación son:

- Recorrido (acceso secuencial)
- Lectura/escritura
- Asignación
- **Actualización (añadir, borrar insertar)**
- Ordenación
- Búsqueda

1. Recorrido (acceso secuencial)

- Se puede acceder a cada elemento de un vector para introducir datos (leer) en él o bien para visualizar su contenido (escribir).
- A la operación de efectuar una acción general sobre todos los elementos de un vector se le denomina recorrido del vector.

1. Recorrido (acceso secuencial)

- Estas operaciones se realizan utilizando estructuras repetitivas, cuyas variables de control (por ejemplo i) se utilizan como subíndices del vector (por ejemplo S[i]).
- El incremento del contador del bucle producirá el tratamiento sucesivo de los elementos del vector.

1. Recorrido (acceso secuencial)

Generalmente se utiliza la estructura de repetición para, ya que se conoce de antemano la cantidad de veces que se desea repetir el bucle:

```
Para i ← 1 hasta n hacer
  escribir('Introduzca el elemento ' ,i, 'del vector F: ')
  leer(F[i])
fin_para
```

1. Recorrido (acceso secuencial)

- También se pueden utilizar las estructuras de repetición mientras y repetir:

```
i ← 1
mientras i <= 20 hacer
   escribir('Introduzca el elemento ' ,i, 'del
                          vector F: ')
   leer(F[i])
   i ← i + 1
fin_mientras
```

1. Recorrido (acceso secuencial)

```
i ← 1
repetir
  escribir('Introduzca el elemento ' ,i, 'del
  vector F: ')
  leer(F[i])
  i ← i + 1
hasta_que i > 20
```

2. Lectura/escritura

La lectura/escritura de datos en arreglos normalmente se realiza con estructuras repetitivas (usando un recorrido secuencial).

Las instrucciones simples de lectura/escritura se representarán como:

- **leer**(A[5]) lectura del elemento 5 del vector A
- **escribir**(A[8]) escribir el elemento 8 del vector A

2. Lectura/escritura

- Para facilitar futuras operaciones con el vector (dependiendo del lenguaje de programación), se puede necesitar inicializar el vector antes de operar con él. Puede usarse cualquier valor que respete el tipo de dato del vector:

```
desde i ← 1 hasta n hacer
  nombre[i] ← '*'
fin_desde
```

3. Asignación

La asignación de valores a un elemento del vector se realizará con la instrucción de asignación:

- A[29] ← 5 asigna el valor 5 al elemento 20 del vector A
- Suma ← A[1] + A[3]
- A[3] ← A[3] + 10.8
- A[1] ← A[4] + A[5]

4. Actualización

La operación de actualización de un vector consta a su vez de tres operaciones más elementales:

- Añadir elementos
- Insertar elementos
- Borrar elementos

4. Actualización

Añadir elementos: es la operación de agregar un nuevo elemento al final del vector. La única condición necesaria para esta operación consistirá en la comprobación de espacio en memoria suficiente para el nuevo elemento.

4. Actualización

Ejemplo: Disponemos de un vector de edades para 7 elementos. Ya hay almacenados 5

EDADES(1), EDADES(2), EDADES(3), EDADES(4) y EDADES(5). Se podrán añadir dos elementos más al final del vector con una simple operación de asignación:

- EDADES(6) $\leftarrow$ 23
- EDADES(7) $\leftarrow$ 20

(Si conoce los espacio del vector que están libres.)

4. Actualización

Si no se sabe si el vector tiene espacios disponibles, primero debe determinarse esto antes de intentar añadir elementos al vector:

```
-- suponiendo vector inicializado a -1
   Para i ← 1 hasta n
     si edades[i]=-1  entonces
       escribir "Introduzca una edad"
       leer edades[i]
     si_no
       cont ← cont + 1
     finsi

     si cont=n
       escribir "El vector no tiene espacio"
     finsi
   finPara
```

4. Actualización

Insertar elementos: consiste en introducir un elemento en el interior de un vector ordenado.

Se necesita un desplazamiento previo para colocar el nuevo elemento en su posición relativa.

Ejemplo: Dado un vector de 8 elementos con nombres ordenados alfabéticamente, se desea insertar dos nuevos nombres: Fernando y Luis.

4. Actualización

1	Ana
2	Carlos
3	Gerardo
4	Lorena
5	Marcos
6	
7	
8	

1	Ana
2	Carlos
3	Fernando
4	Gerardo
5	Lorena
6	Marcos
7	
8	

1	Ana
2	Carlos
3	Fernando
4	Gerardo
5	Lorena
6	Luis
7	Marcos
8	

Como Fernando está entre Carlos y Gerardo se deben desplazar hacia abajo los elementos 3, 4 y 5 que pasarán a ocupar las posiciones relativas 4, 5 y 6.

Posteriormente debe realizarse la misma operación con el nombre Luis que ocupará la posición 6.

Inserción en vector ordenado

```
Proceso Insercion
    definir n,cont, nuevo,i,pos,ocupada como
    entero;
    definir NOMBRES como entero;
n<-10;cont<-0;
Dimension NOMBRES[10];
NOMBRES(1)<-1;
NOMBRES(2)<-2;
NOMBRES(3)<-4;
NOMBRES(4)<-6;
NOMBRES(5)<-7;
NOMBRES(6)<-8;
NOMBRES(7)<-9;
NOMBRES(8)<-10;
NOMBRES(9)<-11;
NOMBRES(10)<-100;
Para i<-1 hasta n Hacer
    ESCRIBIR NOMBRES(i);
FinPara

leer nuevo;//buscamos la posición a insertar
ocupada<-0;
Para i<-1 hasta n hacer
    si (NOMBRES[i]<nuevo) entonces
            cont <- cont + 1 ;
// para determinar la posicion
    finsi
finPara

Pos <- cont + 1; //la posicion del elemento en el vector
cont<-0;
Escribir "La posicion a insertar es ", Pos;

Para i <- 1 hasta n hacer

            si (NOMBRES[i]<>100) entonces
                //posición no vacia
                        ocupada<-ocupada + 1;
            Sino
            cont <- cont + 1; //el numero de posiciones
        libres
            FinSi
finPara
Escribir "Numero de posiciones ocupadas ",ocupada;
Escribir "Valor de contador ", cont;
si cont=0 entonces
            escribir "No se pueden añadir elementos";
Sino
            i<-ocupada;
            Mientras (i >= Pos) hacer
                        NOMBRES[i+1]<-NOMBRES[i];
                        i<- i - 1;
            finMientras
            NOMBRES[Pos]<- nuevo;
            ocupada<- ocupada + 1;
finSi
PARA i<-1 hasta n Hacer
            Escribir NOMBRES(i);
FinPara
FinProceso
```

4. Actualización

Borrar elementos: la operación de borrar el último elemento de un vector no representa ningún problema.

El borrado de un elemento del interior del vector provoca el movimiento hacia arriba de los elementos inferiores a él para reorganizar el vector.

4. Actualización – Borrado en V ordenado

1	Ana
2	Carlos
3	Gerardo
4	Lorena
5	Marcos
6	
7	
8	

1	Ana
2	Carlos
3	Lorena
4	Marcos
5	
6	
7	
8	

Si desea borrar elemento 3 (Gerardo), debe desplazar hacia arriba los elementos de las posiciones 4 (Lorena) y 5 (Marcos).

4. Borrado en V ordenado

Ejemplo: en el vector del ejemplo anterior NOMBRES, borrar el elemento que el usuario desee.

Proceso borrar_elemento
N<-500
Dimension NOMBRES[N]
Escribir 'Introduzca el nombre a borrar:'
leer(nom)

4. Borrado en V ordenado ¿Problemas?

```
j<-1;
Para i ← 1 hasta N
    si (NOMBRES[i]=nom) entonces
       j ← i
    fin_si
  Finpara

  Para i<-j hasta N
     NOMBRES[i] ← NOMBRES[i+1]
Finpara
Finproceso
```

¿Qué ocurre para ultimo elemento? ERROR ACCESO A POSICION NO INICIALIZADA

¿Y si no encontramos el elemento que buscamos? ERROR

Borrado en V ordenado

```
Proceso  borrar_elemento
Definir N,j,i como entero;
Definir NOMBRES como carácter;
Dimension NOMBRES[N];
Definir encontrado como logico;
N<-6;
NOMBRES[1]<-"ANA";
NOMBRES[2]<-"CARLOS";
NOMBRES[3]<-"LAURA";
NOMBRES[4]<-"MARIO";
NOMBRES[5]<-"RAUL";
NOMBRES[6]<-"VICTOR";

Escribir "Introduzca nombre a borrar”;
  leer nom;
  j<-1;
  encontrado<-falso;
Para i<- 1 hasta N Hacer
  si NOMBRES[i]=nom entonces
        j<-i;
       encontrado<-verdadero;
 finsi
Finpara
Si encontrado = verdadero entonces
       Para i<-j hasta N Hacer

          Si i<N entonces
                NOMBRES[i]<-NOMBRES[i+1];
           Sino
            NOMBRES[i]<-" ";
          Finsi
        Finpara
FinSi


Escribir "El vector resultante es";
  Para i<- 1 hasta N Hacer
                Escribir NOMBRES[i];
  Finpara

Finproceso
```