

NOCIÓN DE VECTOR

Nueva estructura de objeto, frecuentemente utilizada en informática y a la que se aplican numerosos tratamientos iterativos. Por ejemplo, supongamos una empresa con 40 sucursales y que deseamos calcular el total de rentas.

El procedimiento a seguir sería ir sumando cada elemento del vector.

$$a) \text{ Total} \Leftarrow \text{Venta}(1) + \text{Venta}(2) + \text{Venta}(3) + \dots + \text{Venta}(40)$$

De esta forma se define el vector en algorítmica a través de un nº índice que va a indicar el elemento del vector con el cual estamos haciendo operaciones.

$$b) \text{ VENTA}(I) \Leftarrow I [1..40]$$

TOTAL \Leftarrow 0 Variable donde vamos a almacenar la sumatoria.

Para I de 1 hasta 40 hacer

TOTAL \Leftarrow TOTAL + VENTA(I)

Fin para

Definición de Vector:

Conjunto ordenado que contiene un nº fijo de elementos (su dimensión) de cualquier tipo válido definido con la condición de que todos deben ser del mismo tipo.

Forma de declarar un vector:

Vector tipo nombre [índice mínimo .. índice máximo]

Ejemplo: Para el ejemplo anterior la declaración del vector quedaría de esta forma:

vector entero venta [1..40]

1.- OPERACIONES CON VECTORES

Asignación

En general, los lenguajes de programación incluyen como muchas instrucciones de lectura/escritura de vectores completos, siendo muy pocos la que tienen definidas otras operaciones primitivas con el vector tomado como dato único.

La estructura de control asociada de manera natural con los vectores es la estructura de tipo "PARA"

Ejemplo:

Algoritmo Inicializar_Vector

Vector de reales V [1 .. 100]

Para i de 1 hasta 100 hacer

$V(i) \leftarrow 0.0$

Fin para

Final

En algunos lenguajes se permite la asignación directa entre vectores de igual dimensión y tipo.

La condición principal para poder realizar este tipo de asignaciones es que debe cumplirse que los dos vectores sean del mismo tipo y de la misma longitud.

$u \leftarrow v$ el vector 'u' pasa a contener los datos de 'v'

Otra forma de hacer esta asignación es a través de la asignación de elemento en elemento. Este tipo de asignación es la que se realiza de forma común en programación, y la que utilizaremos en nuestros algoritmos.

Para i de 1 hasta n hacer

$u(i) \leftarrow v(j)$

Fin para

De este modo veamos un ejemplo: Tenemos dos vectores con las mismas características (tipo y longitud). Se pretende intercambiar los datos entre dos vectores $a \leftrightarrow b$. Para ello utilizaremos un elemento de apoyo c.

Algoritmo Intercambio de vectores

Variables enteras i, n, c

Vectores de enteros a [1..100], b[1..100]

Escribir "Introduzca el número de elementos de los vectores"

Leer n

- Para i de 1 a n

Escribir "Introduzca el elemento a[",i,""]

Leer a[i]

Escribir "Introduzca el elemento b[",i,""]

Leer b[i]

- Fin Para

- Para i de 1 a n

 c \leftarrow a[i] Obsérvese como se realiza el proceso de

 a [i] \leftarrow b[i] intercambio. Necesitamos la variable de

 b [i] \leftarrow c apoyo para no perder por sobrescritura los datos de un vector.

- Fin Para

- Para i de 1 a n

Escribir "a[",i,""]="",a[i]

Escribir "b[",i,""]="",b[i]

- Fin Para

Final

1.1.- NECESIDAD DEL PREDIMENSIONAMIENTO POR EXCESO

Es muy importante entender que la zona de declaración de variables de un algoritmo SIEMPRE SE EJECUTA ANTES QUE LAS SENTENCIAS DEL MISMO. Debido a esto, ES IMPOSIBLE COLOCAR UNA VARIABLE COMO DIMENSION DE UN VECTOR O MATRIZ. Por ello, siempre que pretendamos manipular un número de datos escogido por el usuario, tendremos que recurrir a un predimensionamiento por exceso, considerando un tamaño superior al que éste escogerá. Habitualmente se puede poner 100 como dimensión por defecto, salvo casos en los que se vea claro que sean necesarios más elementos.

2.- LECTURA Y ESCRITURA

Como hemos visto anteriormente, en algorítmica podemos utilizar la asignación de forma directa o por medio de la asignación de los elementos. De igual forma a la hora de leer podemos utilizar la lectura directa o lectura por elementos.

Ejemplo de lectura / escritura por elementos:

Para i de 1 a n hacer

Leer v(i)

Fin para

Para i de 1 a n hacer

Escribir v(i)
Fin para

Finalmente, decir que aunque por facilitar la comprensión de los ejemplos aquí mostrados, se “obviarán” las operaciones de lectura y escritura realizándolas en forma directa, los algoritmos de examen SIEMPRE deberán indicarse estas operaciones desarrolladas, es decir, por elementos.

3.- OPERACIONES ELEMENTALES CON VECTORES NUMÉRICOS

Suma de vectores

Algoritmo suma_vectores

<u>Constante</u> n= ...	⇐	Forma de resolver el problema del
<u>Variable entera</u> i		predimensionamiento por exceso
<u>Vectores reales</u> a[1..n], b[1..n], c[1..n]		
<u>Leer</u> a,b	⇐	LECTURA DIRECTA, NO USAR!
<u>Para</u> i <u>de</u> 1 <u>a</u> n <u>hacer</u>		
c(i) ⇐ a(i)+b(i)		
<u>Fin para</u>		
<u>Escribir</u> c	⇐	ESCRITURA DIRECTA, NO USAR!
<u>Final</u>		

Los vectores que se suman han de ser del mismo tipo y dimensión.

Producto por un escalar

Algoritmo producto por escalar

Constante n= ...
Variable real x
 entera i
vectores reales a(1..n),b(1..n)
Leer a
Leer x
Para i de 1 a n hacer
 b(i) ⇐ x*a(i)
Fin para
Escribir b
Final

Producto escalar de dos vectores

Algoritmo producto escalar

constante n= ...
variable entera i
 real c
vectores reales a(1..n), b(1..n)
Leer a,b
c ⇐ 0.0

```

Para i de 1 a n hacer
    c  $\leftarrow$  c+a(i)*b(i)
Fin para
Escribir c
Final

```

Módulo de un vector

```

Algoritmo modulo
    constante n= ...
    variable entera i
        real modulo, suma_cuadrados
    vector real a (1..n)
    Leer a
    suma_cuadrados  $\leftarrow$  0.0
    Para i de 1 a n hacer
        suma_cuadrados  $\leftarrow$  suma_cuadrados + a(i)*a(i)
    Fin para
    modulo  $\leftarrow$  sqr (suma_cuadrados)
    Escribir módulo
Final

```

4.4.- IDENTIFICACIÓN DE POLINOMIOS CON VECTORES

Sea un polinomio de grado n:

$$P(x) = P_n x^n + P_{n-1} x^{n-1} + \dots + P_1 x + P_0$$

Puede representarse mediante un vector donde sus elementos son los coeficientes del polinomio.

vector real P(0..n)

Suma de polinomios

```

Algoritmo suma_polinomios
    constante n=...
    vectores reales p(0..n),q(0..n),r(0..n)
    variable entera i

    Leer p,q
    Para i de 1 a n hacer
        r(i)  $\leftarrow$  p(i) + q(i)
    Fin para
    Escribir r

```

Final

Evaluación de un polinomio

Regla de Ruffini: valor $p(x)$ para $x=a$. Se basa en dividir el polinomio $p[0..n]$ por $(x-a)$, obteniendo el polinomio cociente, $q[0..n-1]$

$$\begin{aligned}q_{n-1} &= p_n \\q_{n-2} &= p_{n-1} + a q_{n-1} \\&\dots \\q_0 &= p_1 + a q_1 \\r &= p_0 + a q_0\end{aligned}$$

Veamos ahora el algoritmo.

Algoritmo Ruffini

constante $n=...$

Variable real a

entera i

Vectores reales $p(0..n)$, $q(0..n-1)$

Leer p

Leer a

$q(n-1) \leftarrow p(n)$

Para i de $n-1$ a 1 incremento -1 hacer

$q(i-1) \leftarrow p(i) + a * q(i)$

Fin para

$r \leftarrow p(0) + a * q(0)$

Escribir "Polinomio cociente = " q

Escribir "Resto (valor de $p(a)$ = ", r

Final

5.- ORDENACION Y BUSQUEDA

Veamos en este apartado las operaciones comunes en el tratamiento de vectores.

Ordenación por selección

Supongamos un vector, donde parte de sus elementos están ordenados y otros no. Los pasos a seguir para realizar la ordenación son los siguientes:

r	n	t	d	a	h	m	z	o	k	s	c
a ₁	a ₂	...	a _{k-1}	a _k	a _i	a _n

Elementos desordenados

- a) Para realizar la ordenación por selección se hacen varios barridos del vector. Primeramente cogemos el primer elemento del vector y lo comparamos con todos los elementos restantes buscando aquel elemento que posea el valor más pequeño.

r	n	t	d	a	h	m	z	o	k	s	c
a ₁	a ₂	...	a _{k-1}	a _k	a _i	a _n

- b) Una vez encontrado el elemento menor, se coloca en la posición correspondiente, realizando un intercambio de posiciones entre el elemento que hemos comparado y el que hemos encontrado.

a	n	t	d	r	h	m	z	o	k	s	c
a ₁	a ₂	...	a _{k-1}	a _k	a _i	a _n

- c) Una vez ordenado el primer elemento, realizamos la misma operación con los restantes. Finalmente el vector quedará ordenado.

a	c	d	h	k	m	n	o	r	s	t	z
a ₁	a ₂	...	a _{k-1}	a _k	a _i	a _n

El algoritmo de ordenación por selección es el siguiente.

Algoritmo Selección

constante n=...

variable entera i,j,k

real x \leftarrow variable de apoyo

vector de reales a(1..n)

Leer a

Para i de 1 a n-1 hacer

\leftarrow el ultimo 'n' ya estará ordenado.

k \leftarrow i

\leftarrow "k" es el índice del elemento menor a permutar con a[i], inicialmente lo ponemos igual que i

Para j de i+1 a n hacer

si a(j) < a(k) entonces

k \leftarrow j

\leftarrow Si aparece un elemento de menor valor, fijamos "k" a su índice

Finsi

Fin para

x \leftarrow a(i)

\leftarrow Permutamos a[i] con a[k]

a(i) \leftarrow a(k)

a(k) \leftarrow x

Fin para

Escribir a

Final

Ejercicio Propuesto: Realizar la traza para la ordenación por selección del vector a=(75,63,8,15,26,12,2)

Búsqueda Lineal

Dado un vector que suponemos desordenado, localizar un elemento que tenga un valor determinado requerirá un barrido a lo largo del mismo hasta encontrar dicho elemento.

Algoritmo búsqueda_lineal

constante n= ...

variable entera i

real x

vector de reales a(1..n)

Leer a

Leer x

y \leftarrow 1

mientras (a(i) \neq x and i \leq n) hacer

i \leftarrow i+1

fin mientras

si i \leq n entonces

escribir ("Dato ";x;" Encontrado en la posición: ";i)

sino escribir ("Dato no encontrado")

finsi

Final

Búsqueda Binaria

Si el vector está ordenado, existen métodos de búsqueda mas eficientes. Uno de ellos es el de la búsqueda dicotómica. Veamos un ejemplo donde el vector está ordenado de forma ascendente (menor -> mayor).

Algoritmo Busqueda_Dicotómica

constante n=...

variable entera i,j,m

real x

vector a(1..n)

Leer a

Leer x

i \leftarrow 1

j \leftarrow n

repetir

m \leftarrow (i+j)/2

si x < a(m) entonces

j \leftarrow m-1

finsi

si x > a(m) entonces

i \leftarrow m+1

finsi

hasta que (a(m)=x or i>j)

si a(m) = x entonces

escribir ("Dato ";x;" Encontrado en la posición: ";m)

sino escribir ("Dato no encontrado")

finsi

Final

6.- MATRICES.

Podemos extender el concepto de vector a estructuras con dos índices, es decir, matrices.

De forma análoga a la utilización de un bucle, normalmente de tipo "para", en el tratamiento de los elementos de un vector para el caso de una matriz, lógicamente, se requerirán dos bucles compuestos.

Al igual que en el caso de vectores, todos los elementos de una matriz deben ser del mismo tipo. Declaración:

Matriz tipo nombre (índice fila mín..índice fila máx, índice col mín..índice col máx)

Los conceptos de vector y matriz no son sino casos particulares de una estructura general de "p" dimensiones que denominaremos tabla.

7.- OPERACIONES CON MATRICES.

Asignación.

Por ejemplo: inicialización a cero :

```

para i de 1 a m hacer
  para j de 1 a n hacer
    a(i,j)  $\leftarrow$  0.0
  fin para
fin para

```

Definición de matriz identidad

```

para i de 1 a n hacer
  para j de 1 a n hacer
    si i=j entonces
      l(i,j)  $\leftarrow$  1
    sino l(i,j)  $\leftarrow$  0
  finsi
fin para
fin para

```

Asignación de valores introducidos por el usuario:

```

para i de 1 a m hacer
  para j de 1 a n hacer
    leer a(i,j)
  fin para
fin para

```

Al igual que con los vectores, a efectos de simplificar los algoritmos de ejemplo, expresaremos en ocasiones la lectura y escritura de forma abreviada (NO PERMITIDO EL EJERCICIOS DE EXAMEN):

leer a
escribir a

8.- OPERACIONES ELEMENTALES CON MATRICES NUMÉRICAS. SUMA DE MATRICES.

Algoritmo suma de matrices
constantes m=..., n=...
variables enteras i, j
matrices reales a(1..m,1..n), b(1..m,1..n), c(1..m,1..n)
leer a, b
para j de 1 a m hacer
 para i de 1 a n hacer
 $c(i,j) \leftarrow a(i,j) + b(i,j)$
 fin para
fin para

escribir c
Final

Ejercicio Propuesto: Hallar la traza para: a(2,3,-1,4,5,10) y b(1,7,6,18,-3,27)

Evidentemente, las matrices han de ser del mismo tipo y deben tener las mismas dimensiones.

Producto de una matriz por un escalar

Algoritmo producto por un escalar
constantes m=..., n=...
variables enteras i, j
variable real x
matrices reales a(1..m,1..n), b(1..m,1..n)

leer a
leer x

para i de 1 a m hacer
 para j de 1 a n hacer
 $b(i,j) \leftarrow x * a(i,j)$
 fin para
fin para

escribir b
Final

Producto matricial

Algoritmo producto matricial
constantes m=..., n=..., q=...
variables enteras i, j, k
matrices reales a(1..m,1..n), b(1..n,1..q), c(1..m,1..q)

leer a, b
para i de 1 a m hacer
 para j de 1 a q hacer
 c(i,j) \leftarrow 0.0
 para k de 1 a n hacer
 c(i,j) \leftarrow c(i,j) + a(i,k)*b(k,j)
 fin para
 fin para

escribir c
Final

Ejercicios Propuestos

- 1.- Dado un vector "a" de n componentes reales, formular un algoritmo para determinar los componentes máximo y mínimo.
- 2.- Escribir un algoritmo para obtener la traspuesta de una matriz A(m,n).
- 3.- Obtener un algoritmo que lea un conjunto de n datos reales, los almacene en un vector "a" y determine:

a.- El recorrido, $r = \max(a_j) - \min(a_j)$

b.- El valor medio: $\bar{a} = \frac{1}{n} \sum_{i=1}^n a_i$

c.- La desviación típica: $\sigma = \sqrt{\frac{1}{n} \sum_{i=1}^n (a_i - \bar{a})^2}$

d.- El coeficiente de variación: $\frac{\sigma}{a}$

4.- Modificar el algoritmo de ordenación por selección, de forma que opcionalmente la ordenación pueda ser ascendente o descendente.

5.- Sean dos matrices de dimensiones m,n y l,k. Diseñar un algoritmo que detecte aquellos elementos presentes en ambas matrices y los escriba, indicando su posición en las dos matrices (suponiendo que no hay valores repetidos dentro de ninguna de las dos matrices).

6.- Dado un vector de n elementos enteros desordenado, escribir un algoritmo que detecte aquellos elementos que están repetidos, escribiendo el elemento y el número de veces que se repite.