Tema 6. Vectores

1. Introducción y Definiciones
2. Operaciones elementales con vectores
3. Definición y manejo de vectores (arrays) en C
4. Operación de Recorrido secuencial de un vector
5. Operación de Búsqueda secuencial en un vector

Bibliografía:

- Biondi y Clavel, 1989
- Joyanes
- Weiss
- Cairó

6.1. Introducción y Definiciones

Programas =
Algoritmos + Estructuras de Datos

Tipos de datos:

- **Escalares o primitivos.**
  No están formados por tipos de datos más sencillos.
  Ejemplo: numérico, carácter, lógico.

- **Compuestos.**
  Cada elemento de un tipo compuesto está formado por uno o más datos de un tipo escalar o compuestos.
  Ejemplos: vectores, registros, ¿cadenas?

6.1.1. Introducción al tipo Vector

¿Cómo guardar una colección de datos conceptualmente relacionados?

- **almacenar los 100 primeros números primos,**
- **almacenar los divisores de N,**
- **calcular la media de M números y a continuación piden la varianza: ¿es necesario volverlos a leer?**
- **almacenar los 100 primeros números perfectos,...**

Todos los datos son del mismo tipo y están relacionados con el mismo concepto: primos, divisores, perfectos,...

6.1.2. Definiciones

- **Un vector (array o tabla) es un conjunto finito y ordenado de elementos homogéneos (Joyanes, 1997).**
- **Dado un conjunto E con n valores del mismo tipo ($n \geq 1$), un vector V es una aplicación: $V: [1..n] \rightarrow E$ (Biondi y Clavel, 1985)**
- **El número n es el tamaño del vector.**
- **La indexación es la operación elemental que nos permite acceder al i-ésimo valor de E: V(i).**
- **En Informática, un objeto de tipo vector tiene un nombre y un conjunto de valores de un tipo base, a los que se puede acceder mediante un índice.**

<declaración-tipo-vector>::=
<declaración-vector> |
<declaración-matriz>

<declaración-vector> ::=
vector [<rango>] de <tipo>

<rango> ::= <constante>..<constante>

<declaración-matriz> ::=
vector [<rango>] de {(vector [<rango>] de)}
<tipo> |
vector [<rango> {,<rango>}] de <tipo>

Ejemplo 1:
notas: vector[1..6] de numérico;

notas(1), notas(2),..., notas(6) son variables numéricas escalares.

4,5	5,6	7,2	4,0	3,2	8,2
Notas(1)	Notas(2)	Notas(3)	Notas(4)	Notas(5)	Notas(6)

Ejemplo 2:
notas: vector[1..6] de carácter;

notas(1), notas(2),..., notas(6) son variables de tipo carácter.

'S'	'A'	'N'	'S'	'S'	'N'
Notas(1)	Notas(2)	Notas(3)	Notas(4)	Notas(5)	Notas(6)

6.2. Operaciones elementales con vectores

- Asignación
  - Interna:
    notas(1) := 4.5;
    notas(2) := (notas(3) + notas(4)) / 2;
  - Externa:
    Leer notas(j);
    Escribir notas(2);

Ejemplo leer notas de las asignaturas

N: numérico; {parámetro dato; número de datos usados}
notas: vector[1..10] de numérico; {parámetro resultado}
asignatura: vector[1..10] de cadena; {parámetro dato}
Inicio
para i de 1 a N hacer
  escribir "Dame la nota de la asignatura ", asignatura(i);
  leer notas(i)
fin para
Fin

6.3 Definición y manejos de Vectores (arrays) en C

Podemos tener en C vectores de cualquier tipo básico o compuesto:

<tipo> <variable> [<dimension>]⁺;

- Los índices variarán (0..N-1)
- C no comprueba que se exceda el número de elementos del vector
- Los nombres de los vectores (ej. cadena) son equivalente a la dirección en memoria donde comienza el vector (vector[0]).
- &enteros[i] accede a la dirección del elemento i en el vector enteros

```
int enteros[10], matriz[10][20];
char cadena[20];
matriz[9][19] = 12;
cadena[0] = 'c';
for (i=0; i<10; i++) scanf ("%d", &enteros[i])
scanf ("%s", cadena);
```

6.4. Operación de Recorrido Secuencial de un vector

- Operación que consiste en efectuar, sobre TODOS los elementos del vector, una acción genérica que se denomina tratamiento(V(i))
- El recorrido se puede hacer en sentido ascendente o descendente.
- Para recorrer secuencialmente un vector suele utilizarse el esquema para:

```
para I de 1 a N hacer
        tratamiento (V(I));
fin para
```

Ejemplo: calcular la media de las notas leidas antes

```
Inicio
  media:=0;
  para i de 1 a N hacer
     media := media + notas(i);
  fin para
  media := media/N;
Fin
```

6.5. Operación de Búsqueda en un vector

- Problema habitual en programación: buscar un elemento en un vector.
- Tenemos que distinguir dos casos:
  - Los elementos del vector no están ordenados.
    Es necesario recorrer todo el vector para comprobar si un elemento está o no.
  - Los elementos del vector están ordenados.
    Se puede establecer una relación de orden $\leq$ entre los elementos del vector:
    $v(1) \leq v(2) \leq v(3) \leq \ldots \leq v(n)$

    No será necesario recorrer todo el vector en todos los casos.

Búsqueda secuencial en vectores no ordenados

```
Algoritmo Búsqueda_secuencial_1_1 (V, N, X, Posición, Existe) es
V: Vector [1..M] de T; {p. dato}
N: numérico; {p. dato; posiciones ocupadas del vector}
X: T; {p. dato; elemento buscado}
Posición: numérico; {p. resultado; posición del elemento, si está}
Existe: lógico; {p. resultado; indicará si hemos encontrado el
    elemento}
Inicio
 Posición := 1;
 Existe := falso;
 mientras  Posición < N y V(Posicion) <> X hacer
    Posición := Posición + 1;
 fin mientras;
 si V(Posición) = X entonces
   existe := Cierto;
 Fin si
Fin
```

Búsqueda secuencial en vectores no ordenados

```
Algoritmo Búsqueda_secuencial_1_2 (V, N, X, Posición, Existe) es
V: Vector [1..M] de T; {p. dato}
N: numérico; {p. dato; posiciones ocupadas del vector}
X: T; {p. dato; elemento buscado}
Posición: numérico; {p. resultado; posición del elemento, si está}
Existe: lógico; {p. resultado; indicará si hemos encontrado el
    elemento}
Inicio
Posición := 1;
Existe := falso;
mientras  Posición <= N y Existe = falso hacer
  si V(Posición) = X entonces
    existe := Cierto;
  sino
     Posición := Posición + 1;
  fin si
fin mientras;
Fin
```

Búsqueda secuencial en vectores ordenados

```
Algoritmo Búsqueda_secuencial_2 (V, N, X, Posición, Existe) es
V: Vector [1..M] de T;
N, Posición: numérico;
X: T;
Existe: lógico;
Inicio
Posición := 1;
Existe := falso;
mientras Posición < N y V(Posición) < X hacer
        Posición := Posición + 1;
fin mientras;
si V(Posición) = X entonces
        Existe := Cierto;
fin si;
Fin
```

Ejercicios:

1. **Realiza un algoritmo con nombre que lea las *N* notas numéricas del primer cuatrimestre de un *alumno* dado.**

2. **Realiza otro algoritmo con nombre que calcule la media de las *N* notas numéricas de un *alumno* dado.**

3. **Algoritmo que sea capaz de realizar el cambio de cualquiera de las monedas del sistema euro: de la moneda original al euro o viceversa, y entre cualquiera de las monedas.**

4. **Comprobar si un número es primo utilizando la criba de Eratóstenes.**

Ejemplos a resolver:

1. **Utilizando el algoritmo que comprueba si un número es o no primo, realiza un programa que almacene los 100 primeros números primos (en código algorítmico y en C).**

2. **Crea un algoritmo con nombre que calcule cuáles son las notas alfabéticas/simbólicas de un alumno/a a partir de las notas previamente leídas por otro algoritmo en un vector *notas_n*.**

3. **Realiza un algoritmo principal que utilice los dos algoritmos anteriores.**

4. **Implementa el programa anterior y todos los subprogramas en C.**