

TEMA 2

TABLAS

La información utilizable por el computador consiste en una selección de *datos* de la realidad, precisamente el conjunto de datos que se considera relevante para el problema en estudio, y a partir del cual se cree que pueden obtenerse los resultados deseados.

Niklaus Wirth

OBJETIVOS DE ESTE CAPITULO:

Ver el 'array' como la única estructura explícitamente disponible en todos los lenguajes, y como la más simple.

Comprender el comportamiento 'estático' de las tablas, a diferencia del comportamiento dinámico.

Introducir operaciones 'básicas' a realizar con cualquier tipo de estructuras de datos, como búsquedas y ordenaciones.

INDICE TEMA-2

Tablas. 2 horas.

Introducción: Estructura & características

-

Búsqueda Secuencial

- – **Búsqueda Secuencial con centinela**

-

Búsqueda Binaria o Dicotómica

4. Ordenación: selección Directa

6. Ejercicios

1. INTRODUCCIÓN

TABLA: Estructura homogénea

- todos sus componentes son del mismo tipo
- tiene un tamaño predefinido, que no puede variarse en tiempo de ejecución

Declaraciones de tablas:

Estructura Monodimensional:

Type TipoVector = **array** [1..N] **of** TipoElemento;

Estructura Multidimensional:

Type TipoTabla = **array** [1..D1, 1..D2,, 1..Dn] of TipoElemento;

Los elementos de la tabla contienen una clave que los identifica.

Permite acceso directo a cualquiera de sus elementos.

Problema en una tabla:

Localizar la posición del elemento con una clave determinada, conocida a priori.

2. BÚSQUEDA SECUENCIAL

Búsqueda 'exhaustiva' se recorre la tabla, comprobando cada elemento; termina cuando:

1- Encuentra el elemento,

ó

2- Examina todos los elementos de la tabla.

'Programar una búsqueda secuencial de un elemento dentro de una tabla como la que se describe':

Type TipoElemento = **Record**

CampoClave: TipoClave;

OtrosCampos: OtrosTipos

end;

TipoVector = **Array** [1..N] **of** TipoElemento;

Con la siguiente especificación:

Procedure BusquedaSecuencial (Vector; TipoVector;

Clave: TipoClave;

var Elemento: TipoElemento;

var Encontrado: **Boolean**;

var Posicion: **Integer**);

2.1. Búsqueda Secuencial con Centinela

Se incluye en el array (de tamaño N), la posición 'N + 1', llamada **Centinela**, conteniendo la clave del elemento a buscar. Así, se simplifica la condición de fin de búsqueda.

Las declaraciones quedarían:

Type TipoElemento = **Record**

CampoClave: TipoClave;

OtrosCampos: OtrosTipos

end;

TipoVector = **Array**[1..N +1] **of** TipoElemento;

Procedure BusquedaSecuencial (Vector; TipoVector;

Clave: TipoClave;

var Elemento: TipoElemento;

var Encontrado: **Boolean**;

var Posicion: Integer);

var i: Integer; (* N tamaño de la tabla *)

begin

Vector [N + 1].CampoClave := Clave; (* Centinela *)

(* ¿En qué cambiaría, a partir de aquí, el proceso de 'Búsqueda Secuencial' descrito en la página anterior???)

.

.

.3. BUSQUEDA BINARIA o DICOTÓMICA

Se emplea en situaciones donde los elementos están **ORDENADOS** dentro del array.

Se divide sucesivamente en partes iguales el espacio de búsqueda del elemento.

Una tabla se divide en dos, averiguando el centro de la misma:

(Límite inferior + Límite superior) div 2

El problema se reduce entonces a comparar la clave buscada con la del centro de la tabla, para optar por uno u otro intervalo.

Con las declaraciones anteriores, completar el procedimiento de BusquedaBinaria para encontrar la clave 'Clave' dentro de la tabla 'Vector'. La información asociada a la clave buscada ('Clave') ha de devolverse en 'Componente'

Procedure BusquedaBinaria(Vector: TipoVector;

Clave: Tclave;

var Componente: TipoRegistro;

var Posicion: **Integer**;

var Encontrado: **Boolean**);

var Inferior, Superior, Centro: **Integer**; (* N N° máximo de elementos de la tabla *)

begin

Encontrado := **False**;

Inferior := 1;

Superior := N ¿ ! ?

4. ORDENACIÓN DE UN VECTOR: SELECCIÓN DIRECTA

Son muy comunes los métodos de ordenación de tablas, que rehacen la secuencia en que están guardados sus elementos, de acuerdo con alguna relación de orden.

La ordenación puede ser *interna*, si tiene lugar en la memoria del ordenador, y *externa*, si se efectúa sobre elementos almacenados en soportes externos a la misma.

SELECCIÓN DIRECTA: Se encuentra el menor elemento de la tabla (de acuerdo con la relación de orden establecida) y entonces se intercambia con el primero. Este proceso se repite para la sub-tabla que sigue al primer elemento, y así sucesivamente.

Implementar el procedimiento de Selección Directa, de acuerdo con las siguientes declaraciones:

Type TipoVector = **Array** [1.. N] **of** TipoElemento;

Procedure SeleccionDirecta (**var** Tabla: TipoVector);

– 1ª SOLUCION de **Búsqueda secuencial** –

Type TipoElemento = **Record**

CampoClave: TipoClave;

OtrosCampos: OtrosTipos

end;

TipoVector = **Array**[1..N] **of** TipoElemento;

Procedure BusquedaSecuencial (Vector; TipoVector;

Clave: TipoClave;

var Elemento: TipoElemento;

```

var Encontrado: Boolean;

var Posicion: Integer );

(* Búsqueda secuencial de un elemento dentro de una tabla *)

var i: Integer; (* N tamaño de la tabla *)

begin

Encontrado := false;

i := 0;

repeat

i := i + 1;

until ( Vector[ i ].CampoClave = Clave) or ( i < N );

if Vector[ i ].CampoClave = Clave then

begin

Encontrado := true;

Posicion := i;

Componente := Vector[ i ]

end

end; (* Búsqueda Secuencial *)

```

– 2ª SOLUCION de **Búsqueda secuencial** –

```

Type TipoElemento = Record

CampoClave: TipoClave;

OtrosCampos: OtrosTipos

end;

TipoVector = Array[ 1..N ] of TipoElemento;

Procedure BusquedaSecuencial ( Vector; TipoVector;

Clave: TipoClave;

var Elemento: TipoElemento;

```

```

var Encontrado: Boolean;

var Posicion: Integer );

(* Búsqueda secuencial de un elemento dentro de una tabla *)

var i: Integer; (* N tamaño de la tabla *)

begin

i :=1;

Encontrado := false;

while ( i < N ) and ( not Encontrado ) do

if Vector[ i ].CampoClave = Clave then

begin

Componente := Vector[ i ];

Encontrado := true;

Posicion := i;

i := i + 1

end

end; (* Búsqueda Secuencial *)

– Búsqueda secuencial con Centinela

Type TipoElemento = Record

CampoClave: TipoClave;

OtrosCampos: OtrosTipos

end;

TipoVector = Array[ 1..N + 1 ] of TipoElemento;

Procedure BusquedaCentinela ( Vector; TipoVector;

Clave: TipoClave;

var Elemento: TipoElemento;

var Encontrado: Boolean

```

var Posicion: **Integer**);

(* Búsqueda secuencial de un elemento en una tabla. Método del Centinela *)

var i: **Integer**; (* N tamaño de la tabla *)

begin

Vector [N + 1].CampoClave := Clave; (* Centinela *)

i := 0; Encontrado := **false**;

repeat

i := i + 1

until (Vector[i].CampoClave = Clave);

if i > N **then writeln** (`El elemento buscado no está en la tabla')

else begin

writeln (`El índice del elemento es `, i);

Componente := Vector[i];

Encontrado := **true**;

Posicion := i

end

end; (* BusquedaCentinela *)– **Procedimiento de Selección Directa.**

Type TipoElemento = **Record**

CampoClave: TipoClave;

OtrosCampos: OtrosTipos

end;

TipoVector = **Array**[1..N] **of** TipoElemento;

Procedure SeleccionDirecta(**var** Vector: TipoVector);

(* Ordenación por Selección Directa *)

var i, j, k: **Integer**;

X: TipoElemento;

(* N Tamaño del vector *)

begin

for i := 1 **to** N **do**

begin

(* k y X guardan la posición y el contenido del elemento menor *)

k := i;

X := vector [i];

for j := i + 1 **to** N **do**

(* Localización del menor elemento... *)

if Vector [j].CampoClave < X.CampoClave **then**

begin

k := j;

X := Vector [j];

end;

(* ... e intercambio con el primer elemento de la sub-tabla *)

Vector [k] := Vector [i];

Vector [i] := X

end

end; (* SeleccionDirecta *)

Cap. 2 ED-I. TABLAS

Jesús Alonso S. Dpt. OEI p-11-