

TEMA 1

ESTRUCTURAS DE DATOS TIPOS ABSTRACTOS DE DATOS

Practica tu mismo, por dios, con las cosas pequeñas; luego
sigue con las más grandes.

Epíteto, Discursos

OBJETIVOS DE ESTE CAPITULO:

- Un concepto importante en ingeniería: el de Caja Negra
- El rol de las Estructuras de Datos en un programa
- Diferenciar la declaración y las operaciones en una EDs.
- Ventajas de utilizar Estructuras de Datos

INDICE TEMA 1.

1. Definición y propiedades de los TADs. Especificaciones

2. Ejemplos de Especificación de TADs

2.1 TAD Pila de enteros

2.2 TAD Cola de enteros

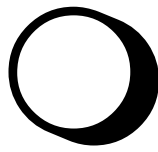
3. Ejemplos de Implementación de TADs

3.1 TAD Pila

3.2 TAD Cola

4. Excepciones

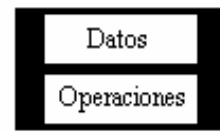
1. Definición y propiedades de los TADs.



Realidad



Abstracción
de la
realidad



Tipo abstracto
de
Datos

ABSTRACCIÓN DE DATOS = { datos, operaciones }

■ tipos de datos: Especificación del TAD

■ operaciones: Implementación del TAD

TAD = ED + Operaciones

➤ **Características de los TADs:**

- ✓ **ENCAPSULAMIENTO:** Se desconoce la implementación de la Declaración y de las Operaciones del TAD.
- ✓ **PROTECCIÓN:** Sólo es posible acceder al TAD a través de las Operaciones del mismo.
- ✓ Representa una **ABSTRACCIÓN:** Se seleccionan ciertos datos que interesan del mundo real, ignorando el resto.
- ✓ Deben poder **COMPILARSE POR SEPARADO.**

➤ **Especificación del TAD:** Permite al usuario la utilización del TAD.

➤ **Implementación del TAD:** Sólo la conoce el programador del TAD, no el usuario.

➤ **Especificaciones**

Sintácticas: Hacen referencia a aspectos Sintácticos:

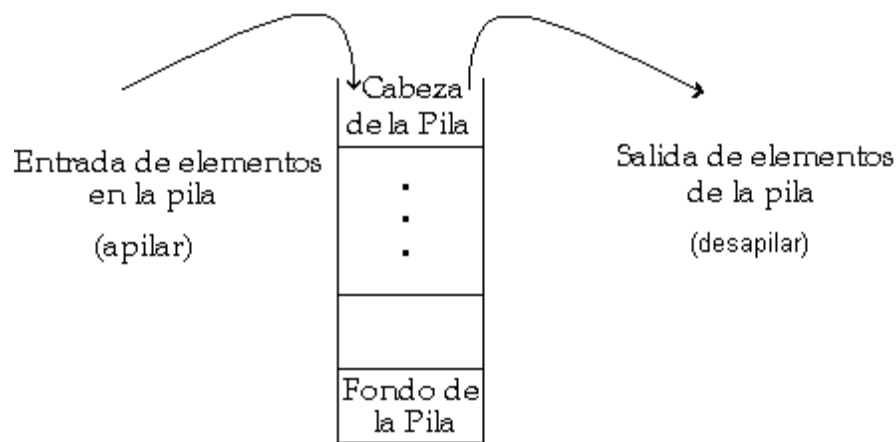
Nombre de la operación, Parámetros de entrada, Parámetros de salida, Tipo de dichos parámetros, Resultado devuelto por la operación, Tipo del resultado, etc.

Semánticas: Indican el efecto producido por la operación.

Para CADA OPERACIÓN del TAD es necesario dar sus especificaciones Sintácticas y Semánticas.

2. Ejemplo de Especificación de TAD: Pila de enteros.

- Una Pila es una ED que almacena elementos de un tipo determinado, que sigue la política de que el primer elemento que se extrae (se ‘desapila’) es el último que se introdujo (se ‘apiló’) -primero en salir, ultimo en entrar (LIFO)-.



TAD Pila

- ❏ Todos los procesadores tienen un mecanismo de este tipo conocido por el nombre ‘Pila del Sistema’, el cual se utiliza para guardar los parámetros de las subrutinas, y otros valores.

ESPECIFICACIÓN del TAD PILA

➤ **Operación: Inicializar una pila.**

Especificación Sintáctica:

- ❶ Nombre de la operación: INIPILA.
- ❷ Parámetros de entrada, y tipos: no tiene
- ❸ Parámetros de salida: una pila, de tipo TPila.
- ❹ Resultado: ninguno.
- ❺ Declaración: **Procedure** IniPila (**var** Pila: TPila);

Especificación Semántica:

Crea una pila, que inicialmente está vacía.

➤ **Operación: Apilar un elemento en la pila**

Especificación Sintáctica:

- 1 Nombre de la operación: APILAR
- 2 Parámetros de entrada, y tipos:
 - Pila que recibe al elemento, de tipo TPila.
 - Elemento que se introduce en la pila, de tipo Entero.
- 3 Parámetros de salida: la propia pila, de tipo TPila, con el elemento introducido.
- 4 Declaración: **Procedure** Apilar (Elemento: **Integer**;
var Pila: TPila);

Especificación Semántica:

Introduce en la pila el elemento pasado como parámetro. La próxima vez que se ejecute la operación ‘DESAPILAR’, el elemento extraído será el ahora introducido.

➤ Operación: Desapilar un elemento de la pila

Especificación Sintáctica:

❶ Nombre de la operación: DESAPILAR

❷ Parámetros de entrada, y tipos:

Pila que contiene el elemento, de tipo TPila.

❸ Parámetros de salida: la propia pila, con el elemento introducido.

Pila de la que se ha extraído el elemento, de tipo TPila.

Elemento que se extrae de la pila, de tipo Entero.

❹ Declaración: **Procedure** Desapilar (**var** Pila: TPila;
var Elemento: Integer);

Especificación Semántica:

Extrae de la pila un elemento. El elemento extraído es justamente el último que se introdujo.

➤ Operación: Determinar si la pila está vacía.

Especificación Sintáctica:

❶ Nombre de la operación: PILAVACIA

❷ Parámetros de entrada, y tipos:

Pila que se quiere comprobar, de tipo TPila.

❸ Parámetros de salida: ninguno.

❹ Resultado: un valor booleano.

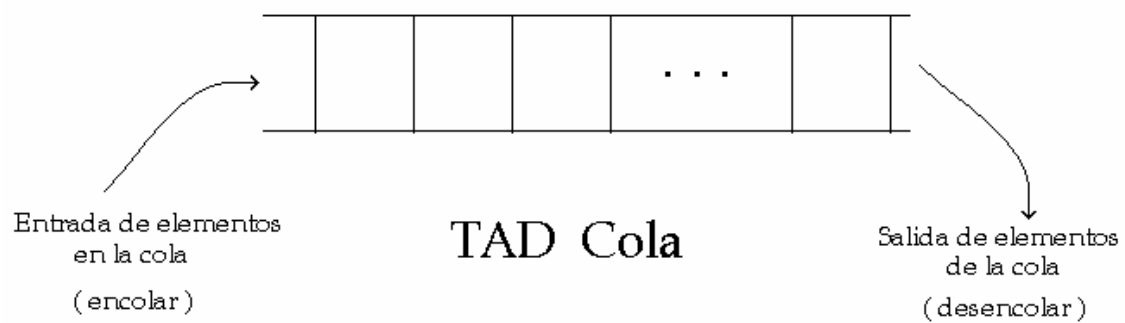
❺ Declaración: **Function** PilaVacía (Pila: TPila): **Boolean**;

Especificación Semántica:

Devuelve el valor 'cierto' si la pila no contiene ningún elemento.

3. Ejemplo de Especificación de TAD: Cola de enteros.

- Una Cola es una ED que almacena elementos de un tipo determinado, que sigue la política de que el primer elemento que se extrae (se ‘desencola’) es el primero que se introdujo (se ‘encoló’) -primero en salir, primero en entrar (FIFO)-.



ESPECIFICACIÓN del TAD COLA

➤ Operación: Inicializar una cola

Especificación Sintáctica:

- ❶ Nombre de la operación: INICOLA.
- ❷ Parámetros de entrada, y tipos: no tiene
- ❸ Parámetros de salida: una cola, de tipo TCola.
- ❹ Declaración: **Procedure** IniCola (var Cola: TCola);

Especificación Semántica:

Crea una cola, que inicialmente está vacía.

➤ Operación: Cola Vacía

Especificación Sintáctica:

- ❶ Nombre de la operación: COLAVACIA.
- ❷ Parámetros de entrada, y tipos: una cola perteneciente al tipo TCola.
- ❸ Parámetros de salida: ninguno.
- ❹ Declaración: **Function** ColaVacia (Cola: TCola): **boolean**;

Especificación Semántica:

Devuelve ‘verdad’ si la cola pasada como parámetro está vacía; ‘falso’ en caso contrario.

➤ Operación: Encolar un elemento en la cola

Especificación Sintáctica:

- ❶ Nombre de la operación: ENCOLAR

② Parámetros de entrada, y tipos:

Cola que recibe al elemento, de tipo TCola.

Elemento que se introduce en la pila, de tipo Entero.

③ Parámetros de salida: la propia cola, de tipo TCola, con el elemento incluido.**④ Declaración: **Procedure** Encolar (Elemento: **Integer**;
var Cola: TCola);**

Especificación Semántica:

Introduce en la cola el elemento pasado como parámetro. Este elemento será el último en salir, de todos los elementos presentes en este momento.

➤ Operación: Desencolar un elemento de la cola.

Especificación Sintáctica:

① Nombre de la operación: DESENCOLAR.**② Parámetros de entrada, y tipos:**

Cola que contiene el elemento, de tipo TCola.

③ Parámetros de salida: la propia cola, con el elemento incluido.

Cola de la que se ha incluido al elemento, de tipo TCola.

Elemento que se extrae de la cola, de tipo Entero.


④ Declaración: **Procedure Desencolar (var Cola: TPila;
var Elemento: Integer);**


Especificación Semántica:


Extrae de la cola un elemento. Dicho elemento es el primero que se introdujo, entre los que se encuentran actualmente en la cola.


4. Implementación del TAD Pila.

```
Type Contenido = Array [ 1.. N ] of Integer; (* N  $\rightarrow$  tamaño de la pila *)  
    TPila = record  
        PP: Integer; (* Índice de la Pila *)  
        Contenido: TContenido  
    end;
```

```
 Function PilaVacía ( Pila: TPila ): Boolean;  
    begin  
        PilaVacía := ( Pila.PP = 0 )  
    end;
```

```
 Procedure Inicializar ( var Pila: TPila );  
    begin  
        Pila.PP := 0  
    end;
```

```
 Procedure Apilar ( var Pila: TPila; Elemento: Integer );  
    begin  
        with Pila do  
            if PP < N then  
                begin  
                    PP := PP + 1;  
                    Contenido[ PP ] := Elemento  
                end  
        end;  
    end;
```

```
 Procedure DesApilar ( var Pila: TPila; Elemento: Integer );  
    begin  
        with Pila do  
            if PP > 0 then  
                begin  
                    Elemento := Contenido[ PP ];  
                    PP := PP - 1  
                end  
        end;  
    end;
```

EJEMPLO DE UTILIZACION DE UN TAD

Implementar las operaciones que se realizan sobre un array de enteros, utilizando exclusivamente el TAD Pila. Se pide:

1. Implementar un procedimiento que efectue la operación de reservar espacio para un array de N elementos.
2. Realizar una función que recupere el contenido de una determinada posición del array.
3. Escribir un procedimiento que simule la operación de escribir un dato en una determinada posición del array.

IMPORTANTE: Al utilizar un TAD, para acceder a la estructura de datos oculta en él, sólo es posible a través de los procedimientos definidos en dicho TAD.


4. Implementación del TAD Cola.

Type TCola = record

Primero, Ultimo, NumElementos: **Integer**;

Contenido: Tcontenido


end;

 **Function ColaVacia (Cola: Tcola): Boolean;**

begin

ColaVacia := (Cola.NumElementos = 0)


end;

 **Procedure IniCola (var Cola: Tcola);**

begin

Cola.Numelementos := 0

end;

 **Procedure Encolar (var Cola: TCola; Elemento: Integer);**

begin

with Cola do

if NumElementos < N then

begin

NumElementos := NumElementos + 1;

if NumElementos = 1 then (* La cola está vacía *)

begin

Primero := 1;

Ultimo := 1

end

else if ultimo = N (* La cola está llena *)

then Ultimo := 1

else Ultimo := Ultimo + 1;

Contenido [Ultimo] := Elemento

end

end;

```
Procedure DesEncolar ( var Cola: Tcola; var Elemento: Integer);  
  begin  
    with Cola do  
      if NumElementos > 0 then  
        begin  
          NumElementos := NumElementos - 1;  
          Elemento := Contenido [ Primero ];  
          If Primero = N  
            then Primero := 1  
            else Primero := Primero - 1  
        end  
      end  
    end;
```

6. Excepciones

A la hora de especificar un TAD, puede ocurrir que esa operación no pueda efectuarse en los términos expresados. Estas circunstancias dan origen a ‘Excepciones’, que abrán de acompañar, si existen, a las Especificaciones Sintácticas y Semánticas.