

Universidad Zaragoza
1474

Profesores:
Alfonso López Ruiz

Práctica 4
Programación de TAD genérico arborescente en C++

Estructuras de datos y algoritmos

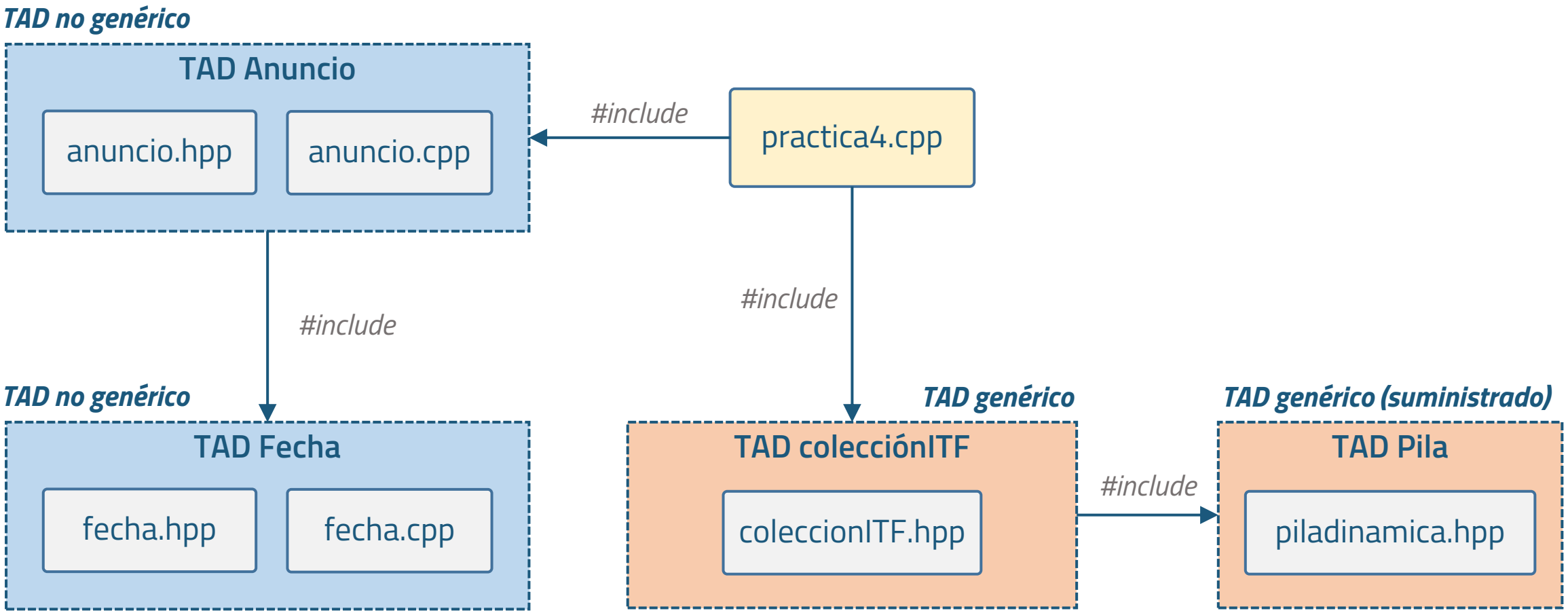
Grado en Ingeniería Informática – Curso académico 2026-2027

CC BY NC ND

OBJETIVOS DE LA PRÁCTICA

- Implementar un **TAD genérico en memoria dinámica con estructura arbórea**.
  - Colección de ítems temáticos fechados.
  - **Árbol binario de búsqueda**.
- Implementar un **TAD no genérico**.
  - **Anuncio.**
- Comprobar de manera automática el funcionamiento del TAD genérico.
  - **Leer instrucciones y datos** de fichero.
  - **Construir un fichero de salida** con los resultados.

ESQUEMA DE LA SOLUCIÓN



TAD NO GENÉRICO Y GENÉRICO EN C++

TAD no genérico: anuncio

```
// anuncio.hpp
struct anuncio;
string suTitulo(const anuncio& e);
...

struct anuncio {
    friend string suTitulo(const anuncio& e);
    ...
  private:
    string titulo;
    ...
};

// anuncio.cpp: implementación
string suTitulo(const anuncio& e) { ... }
```

TAD genérico: colecciónITF<T>

```
// coleccionITF.hpp
template<typename T> struct coleccionITF;
template<typename T> int tamanyo(const coleccionITF<T>& c);
...

template<typename T>
struct coleccionITF {
    friend int tamanyo<T>(const coleccionITF<T>& c);
    ...
  private:
    ...                          // la representación
};

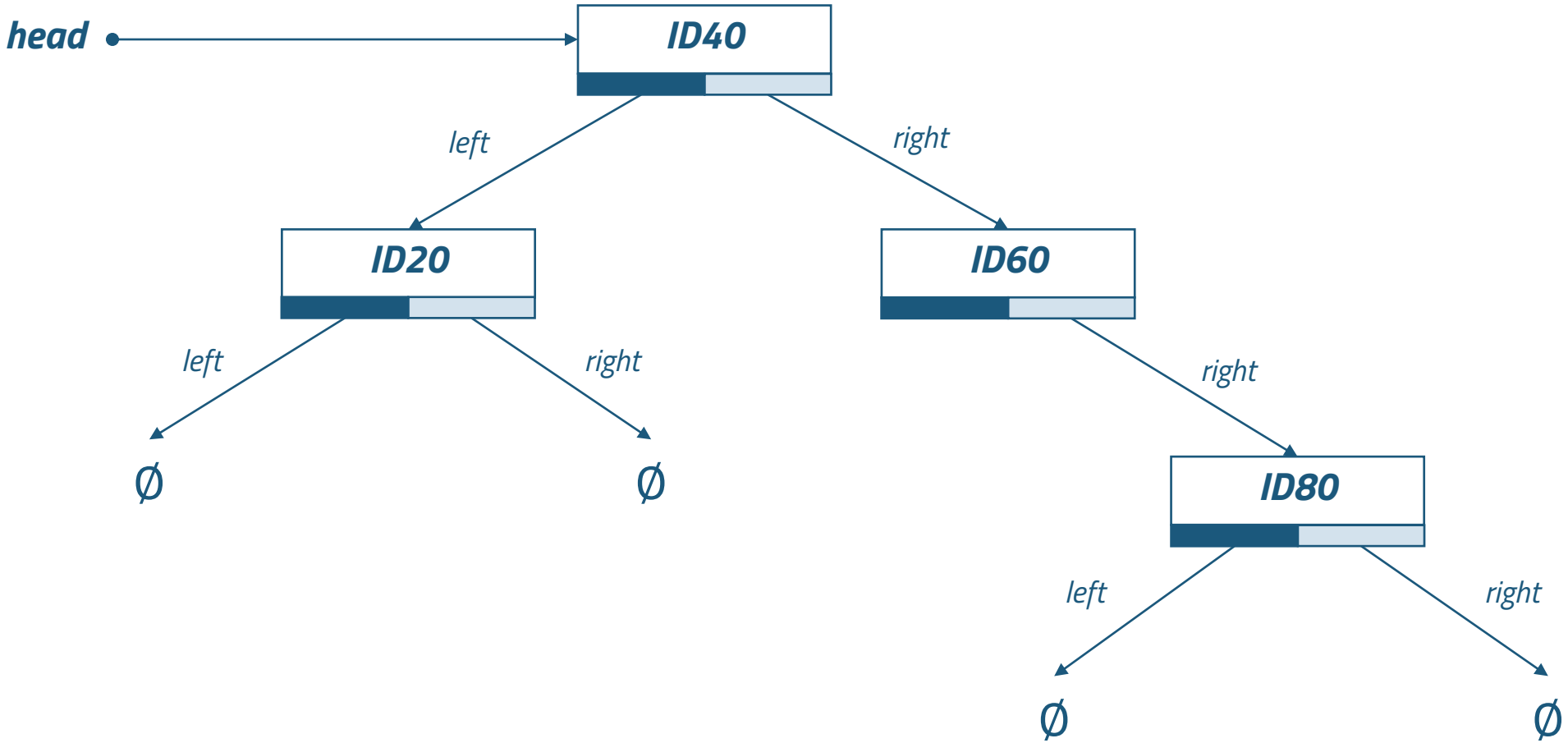
template<typename T>
int tamanyo(const coleccionITF<T>& c) { ... }
```

ESPECIFICACIÓN DEL TAD GENÉRICO *COLECCIÓN ITF*


```
espec coleccionesITF
usa booleanos, naturales, cadenas, fechas
parámetro formal
 género ITF {Ítem Temático Fechado}
 operación
 {Se requiere que el género ITF tenga definidas:}
      identificador: ITF i -> cadena
      {devuelve una cadena con el identificador del ítem i.}
      tema: ITF i -> cadena
      {devuelve una cadena con el tema del ítem i.}
      fecha: ITF i -> fecha
      {devuelve la fecha perteneciente al ítem i.}

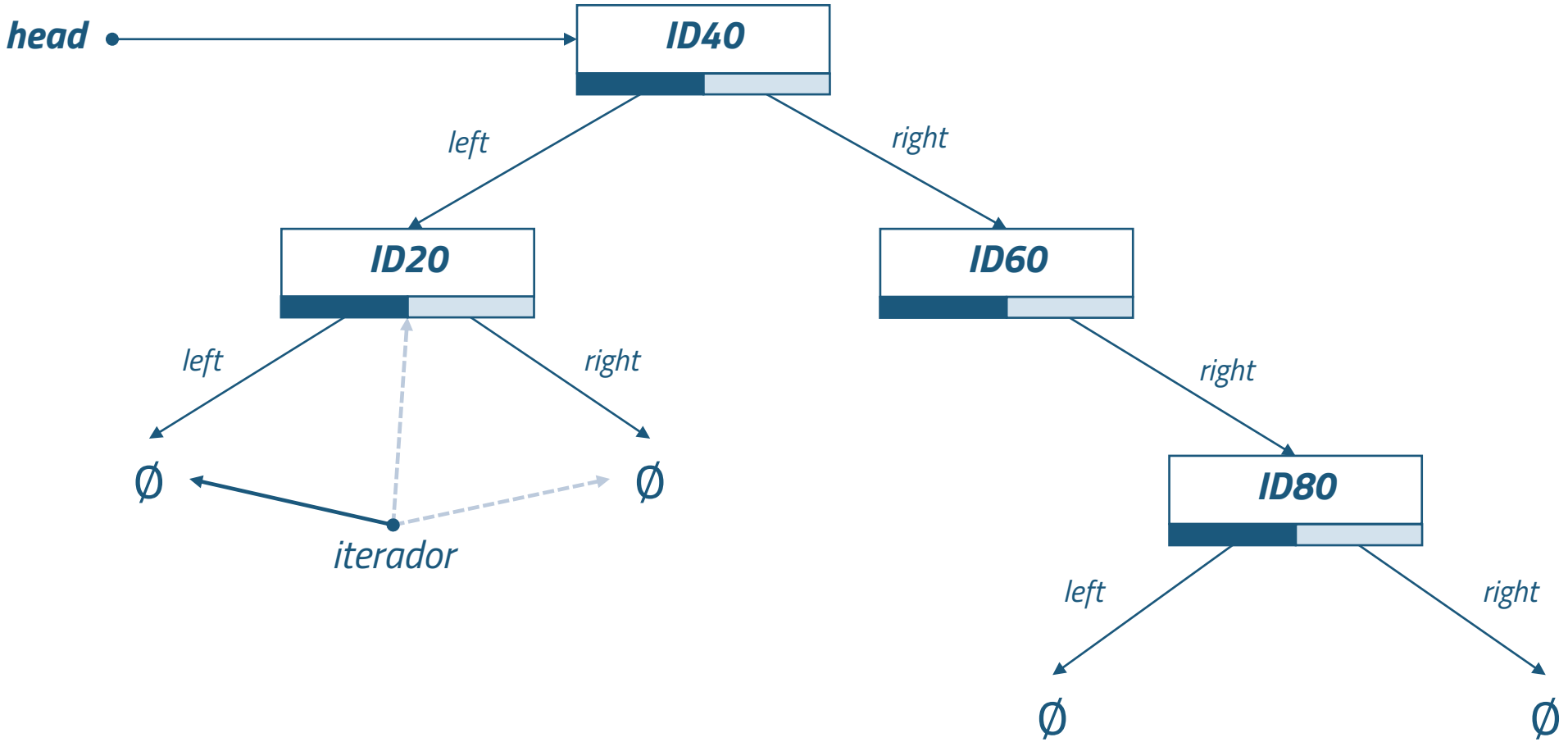
fpf
```

ESPECIFICACIÓN DEL TAD GENÉRICO *COLECCIÓN ITF*



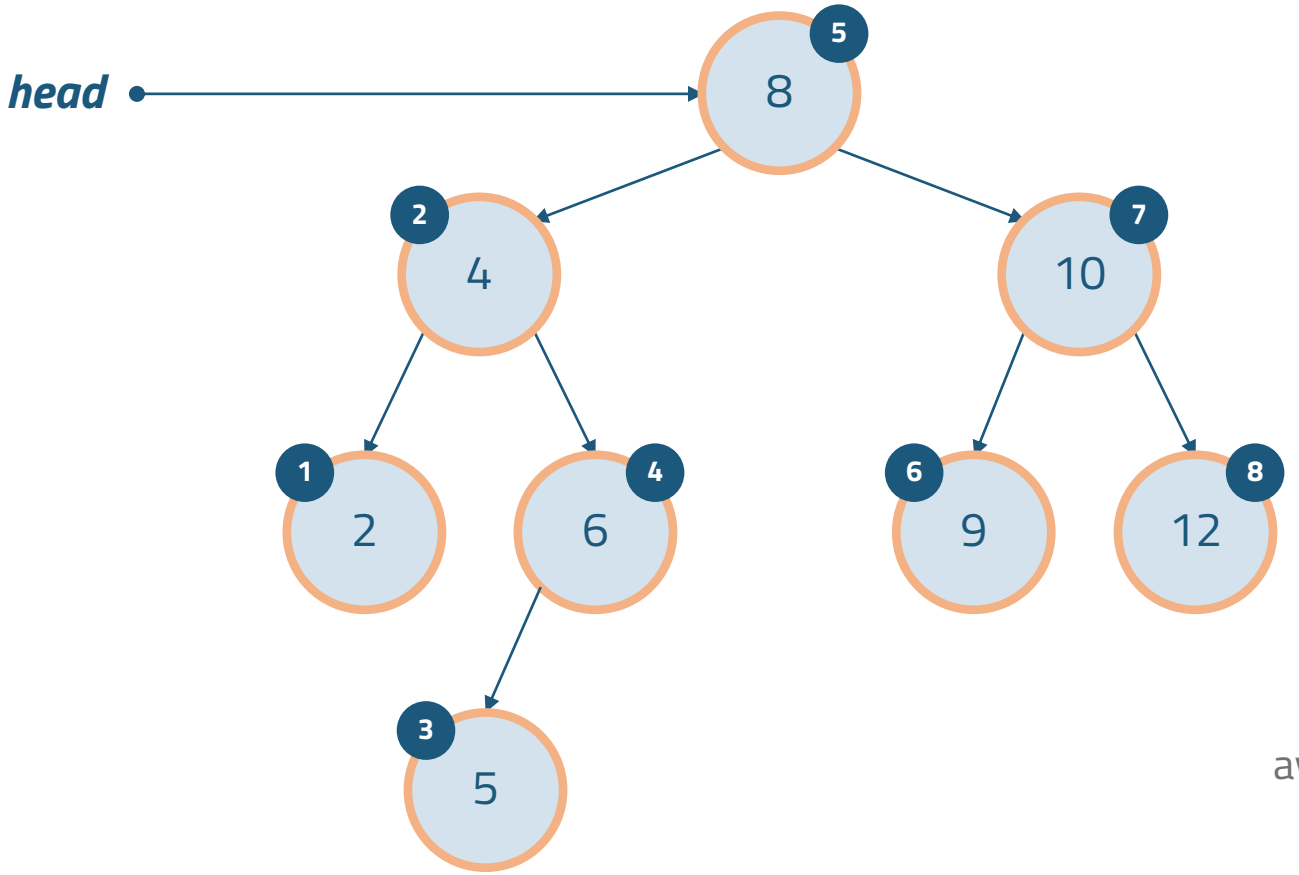
ESPECIFICACIÓN DEL TAD GENÉRICO *COLECCIÓN ITF*


```
operaciones
     iniciarIterador: colecciónITF c -> colecciónITF
     {Inicializa el iterador para recorrer los ítems de c, de forma que el siguiente ítem
      a visitar sea el de identificador (lexicográficamente) menor.}
```



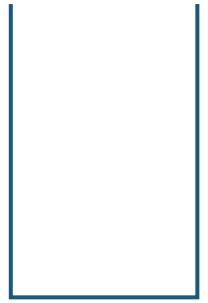
ESPECIFICACIÓN DEL TAD GENÉRICO *COLECCIÓN ITF*

```
operaciones
     iniciarIterador: coleccionITF c -> coleccionITF
     {Inicializa el iterador para recorrer los ítems de c, de forma que el siguiente ítem
      a visitar sea el de identificador (lexicográficamente) menor.}
```



Inorden

izquierdo → raíz → derecho



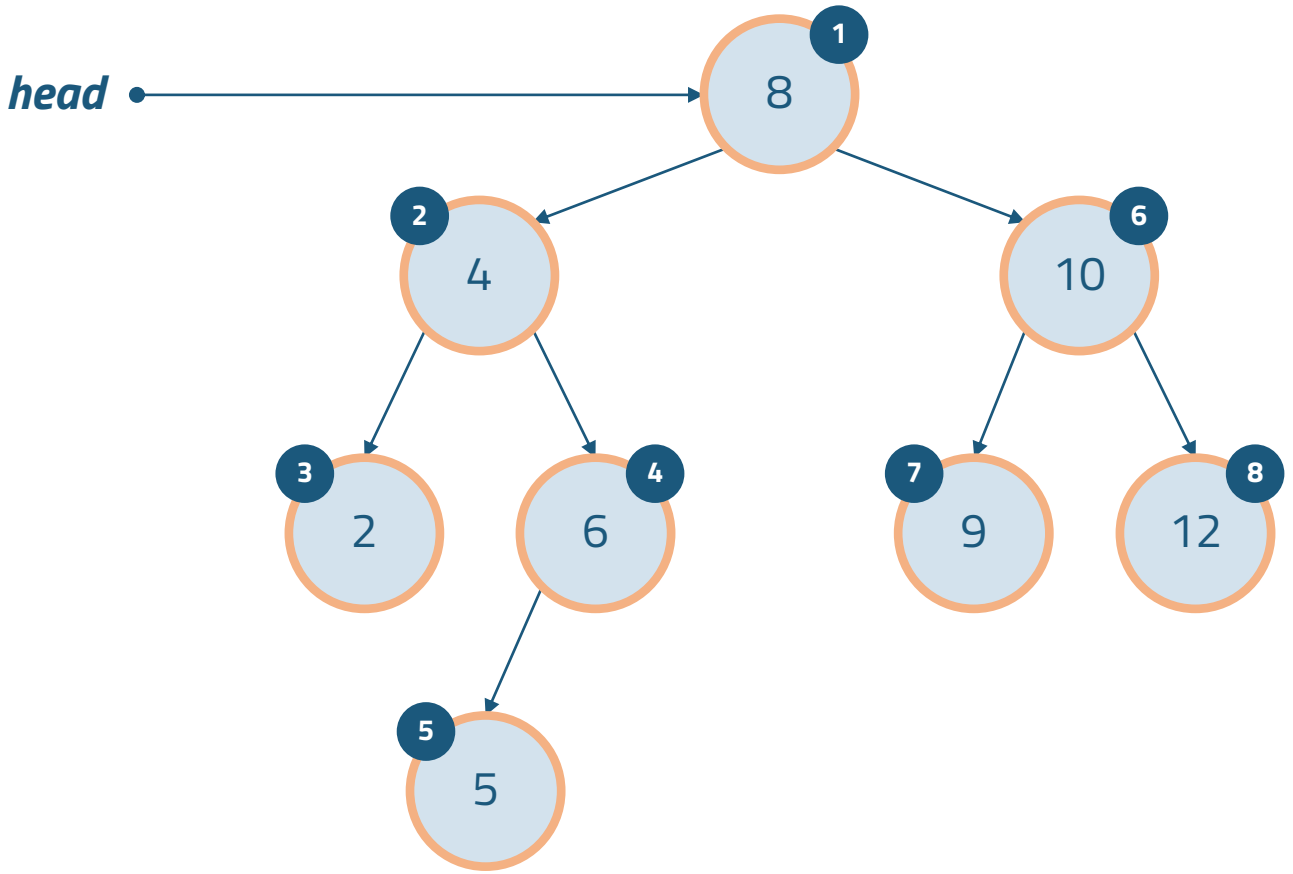
pila

`pop()` → **12**

avanza: la pila queda vacía, fin del recorrido

ESPECIFICACIÓN DEL TAD GENÉRICO *COLECCIÓN ITF*


```
operaciones
     delTema: cadena t, cadena separador, coleccionITF c -> cadena
     {Concatena los identificadores de los ítems de c con tema igual a t, separados
      entre sí por la cadena separador.}
```



Preorden

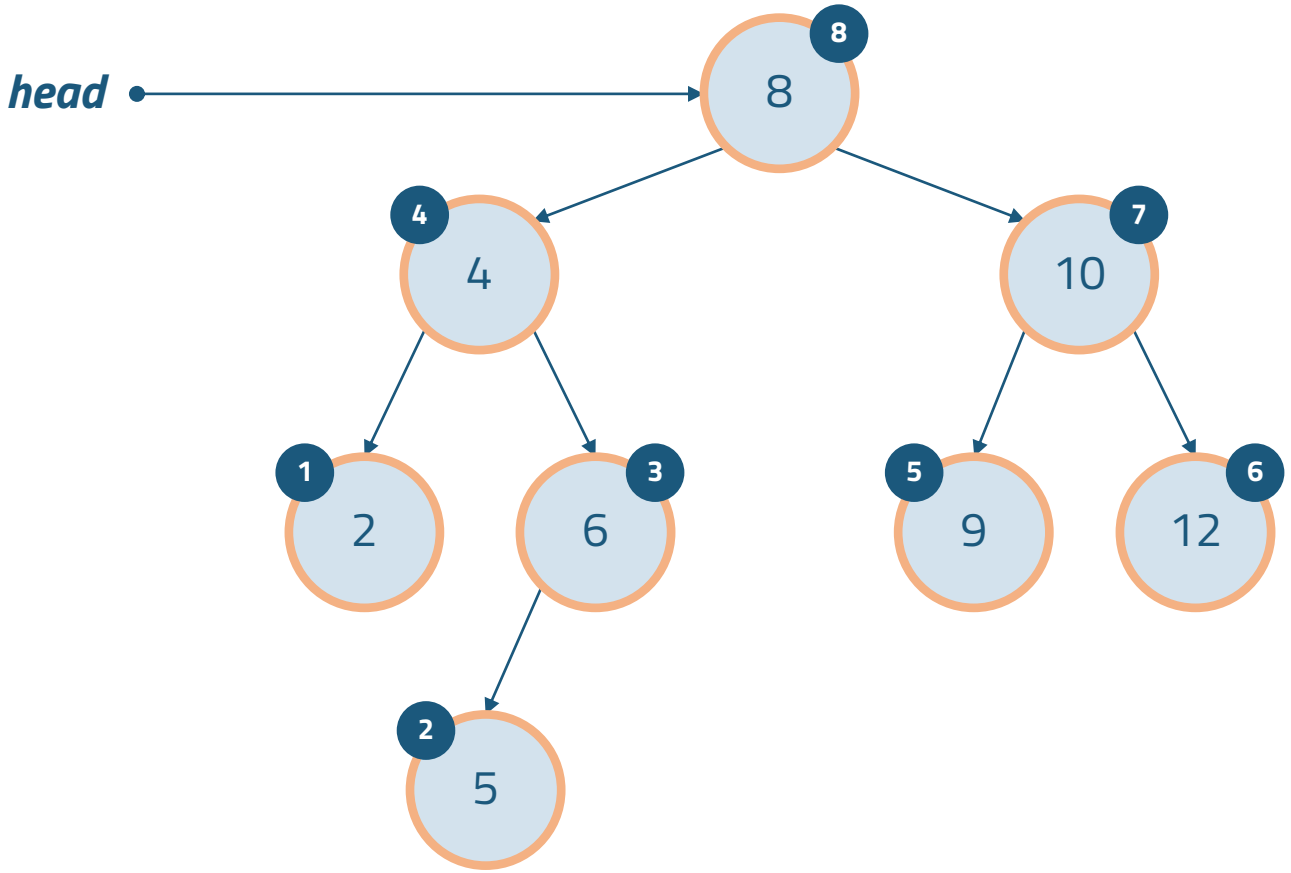
raíz → izquierdo → derecho

orden de visita

8	4	2	6	5	10	9	12

ESPECIFICACIÓN DEL TAD GENÉRICO *COLECCIÓN ITF*


```
operaciones
     obsoletos: fecha f, cadena separador, coleccionITF c -> cadena
     {Concatena los identificadores de los ítems de c con fecha anterior a f, separados
      entre sí por la cadena separador.}
```



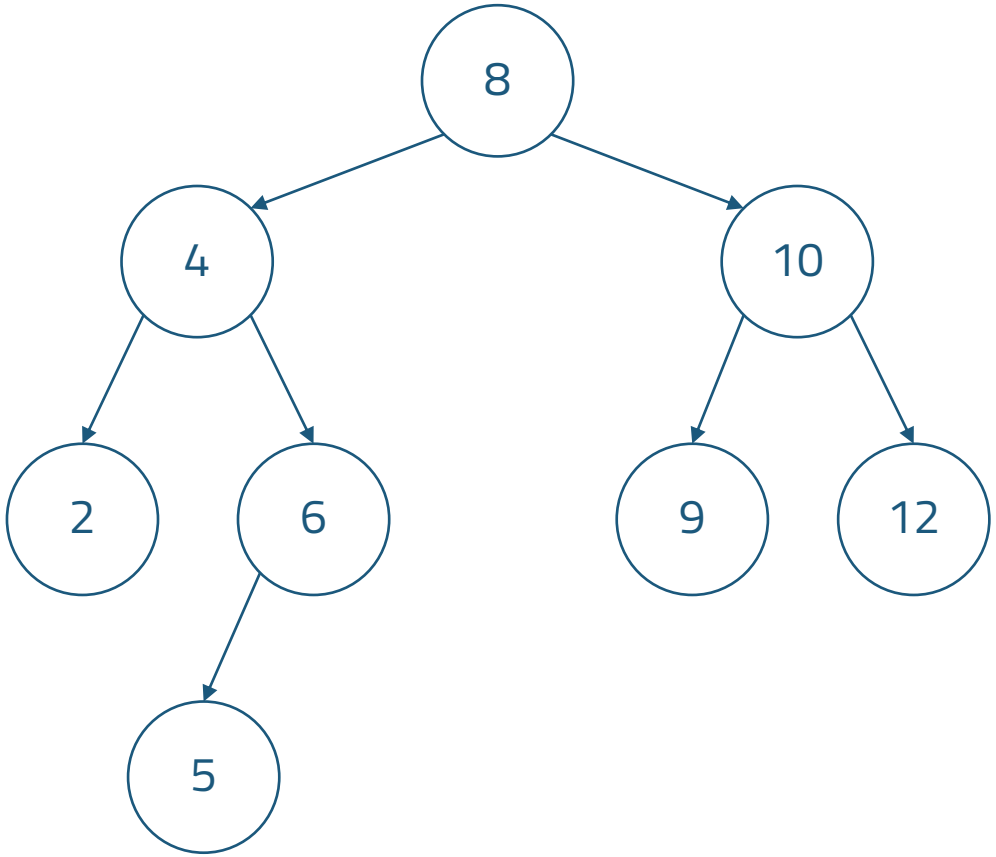
Postorden

izquierdo → derecho → raíz

orden de visita

2	5	6	4	9	12	10	8

RECORRIDOS DEL ÁRBOL



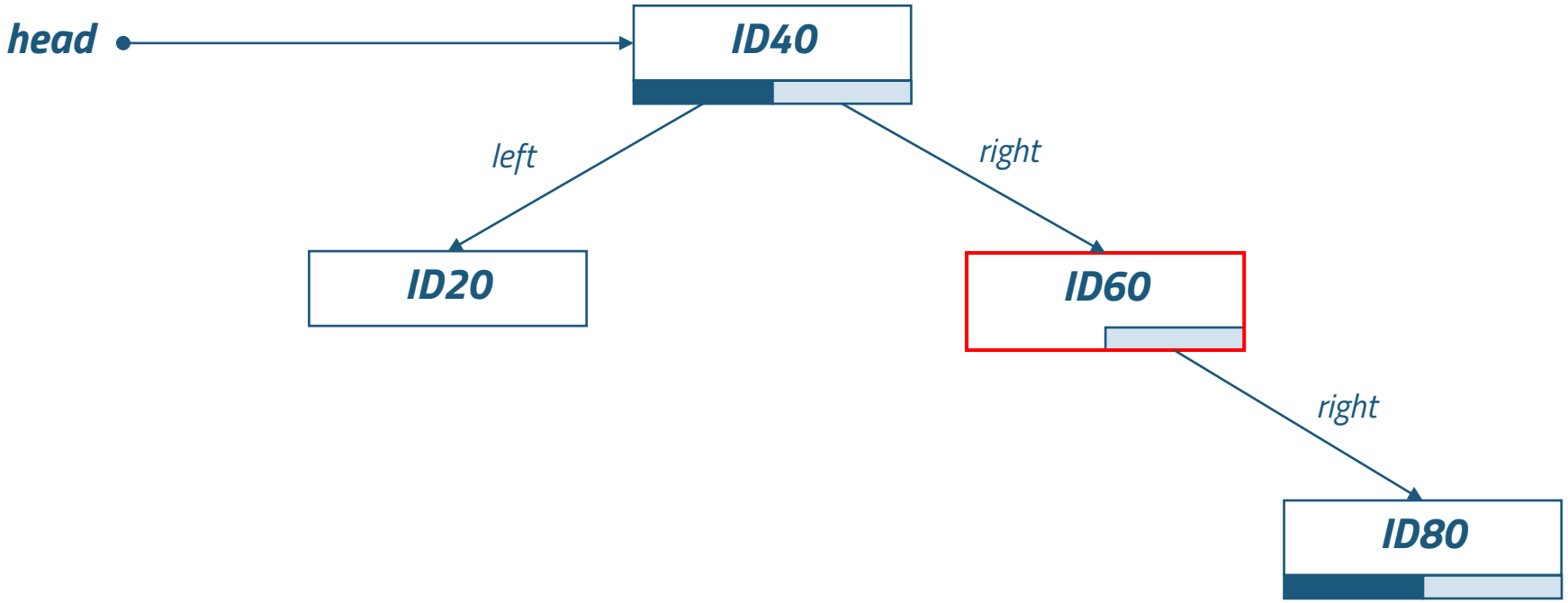
Recorrido	Orden de visita	Operación
Inorden izquierdo → raíz → derecho	2 4 5 6 8 9 10 12	iterador (LT)
Preorden raíz → izquierdo → derecho	8 4 2 6 5 10 9 12	delTema (LC)
Postorden izquierdo → derecho → raíz	2 5 6 4 9 12 10 8	obsoletos (LO)

ESPECIFICACIÓN DEL TAD GENÉRICO *COLECCIÓN ITF*

operaciones

borrar: cadena ident, colecciónITF c -> colecciónITF

{Si en c hay algún ítem con identificador igual a ident, lo borra. Si no, devuelve c.}

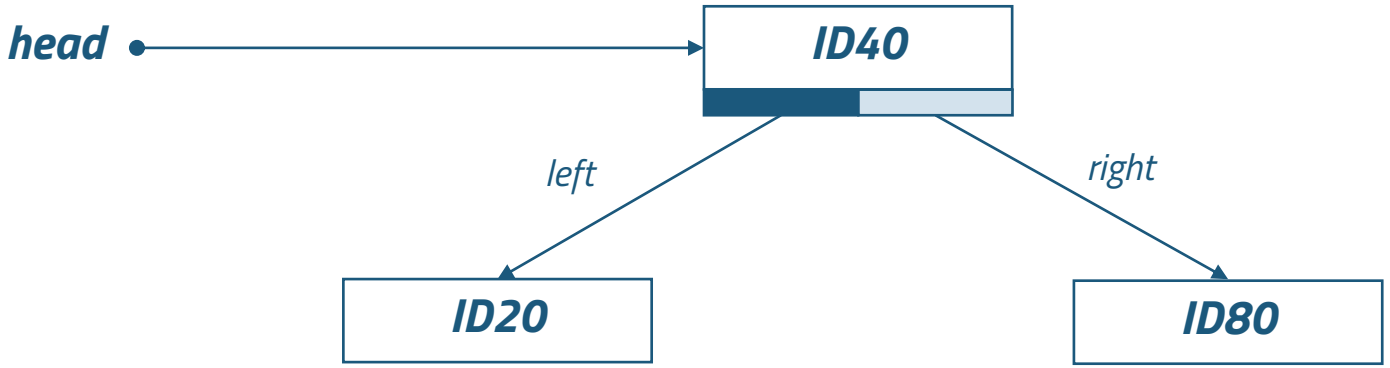


ESPECIFICACIÓN DEL TAD GENÉRICO *COLECCIÓN ITF*

operaciones

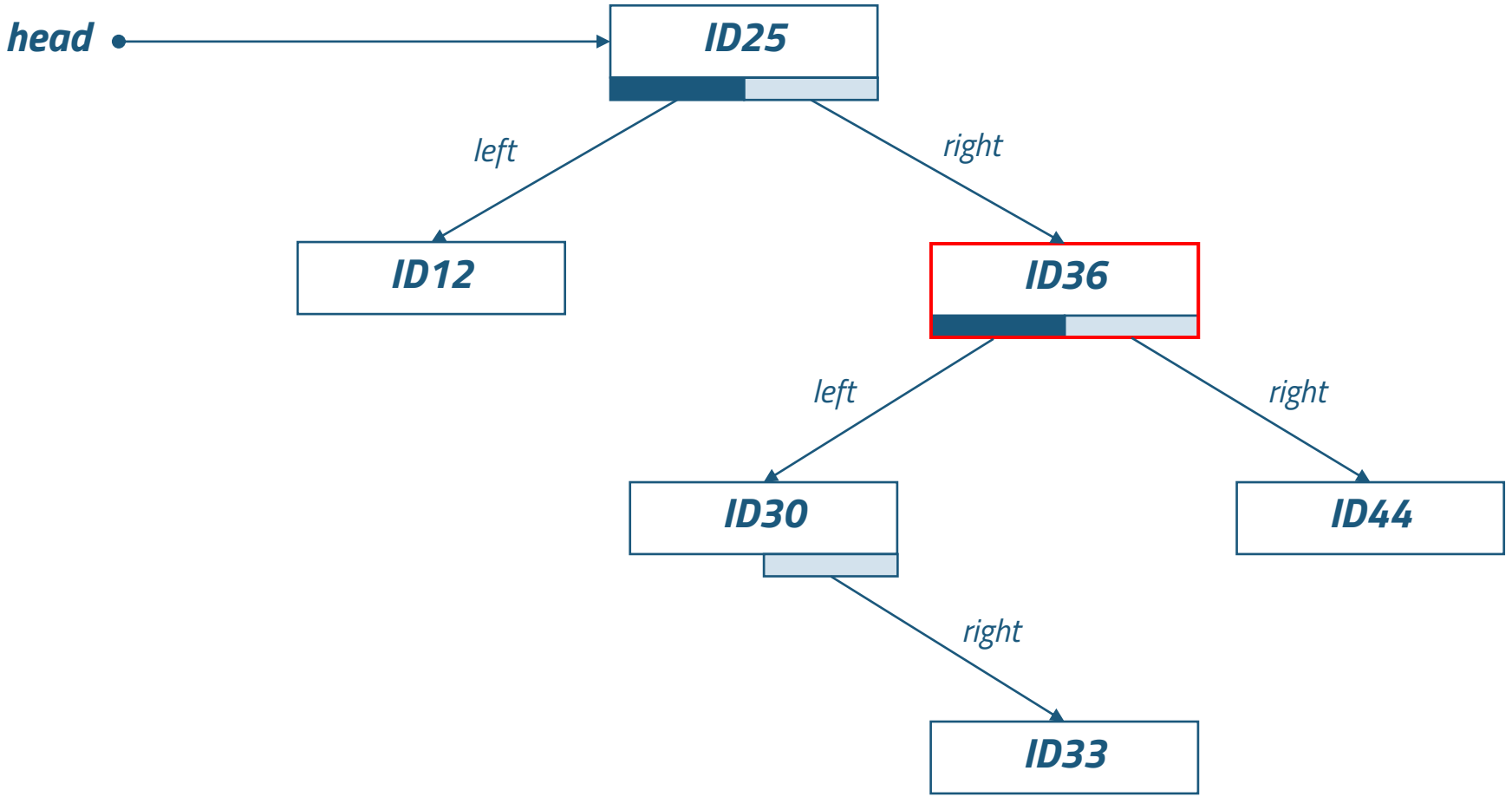
borrar: cadena ident, coleccionITF c -> coleccionITF

{Si en c hay algún ítem con identificador igual a ident, lo borra. Si no, devuelve c.}



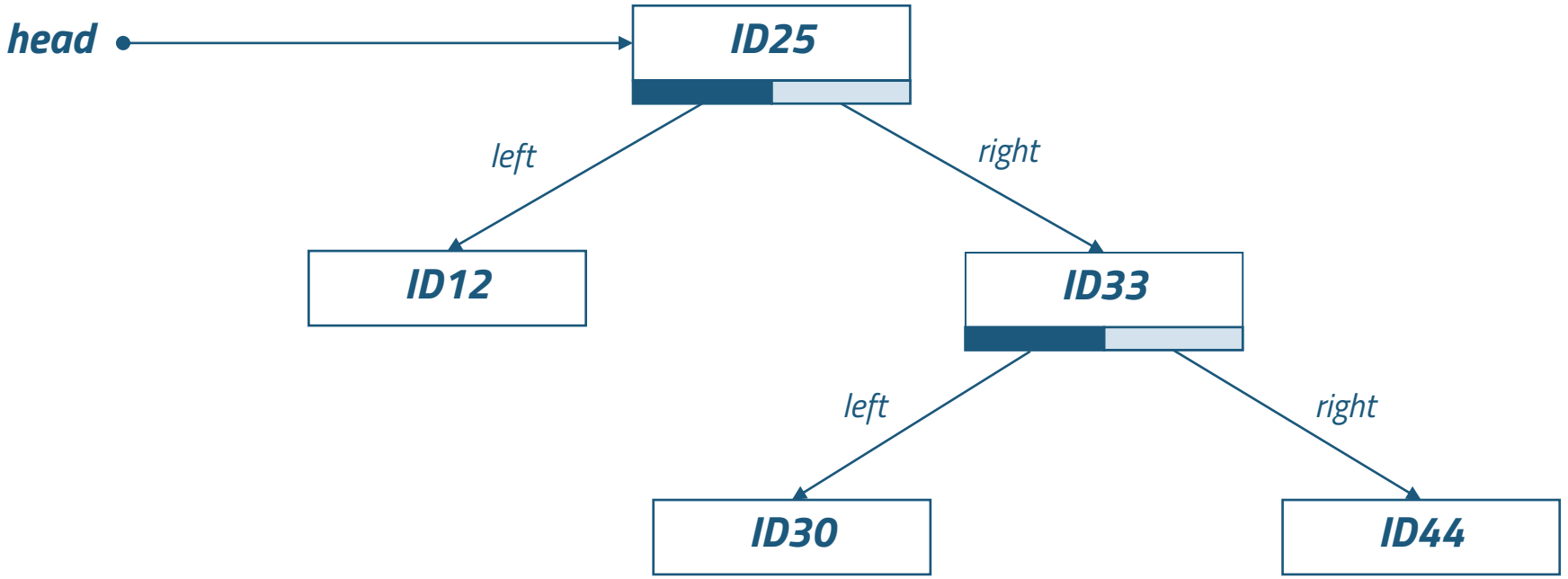
ESPECIFICACIÓN DEL TAD GENÉRICO *COLECCIÓN ITF*

```
operaciones
      borrar: cadena ident, coleccionITF c -> coleccionITF
      {Si en c hay algún ítem con identificador igual a ident, lo borra. Si no, devuelve c.}
```



ESPECIFICACIÓN DEL TAD GENÉRICO *COLECCIÓN ITF*

```
operaciones
      borrar: cadena ident, colecciónITF c -> colecciónITF
      {Si en c hay algún ítem con identificador igual a ident, lo borra. Si no, devuelve c.}
```



ENTREGA DE LA PRÁCTICA

- Comprobar resultados en **lab000**.
  - **g++ -std=c++11 *.cpp**.
- Utilizar siempre ***"entrada.txt"*** y ***"salida.txt"*** como rutas.
- **La salida debe coincidir exactamente** con la exigida en el guion.
  - Mayúsculas, minúsculas, espacios, caracteres, etc.
- **No** se pueden utilizar **excepciones** ni **smart pointers**.
- **No** se puede utilizar **Programación Orientada a Objetos** ni **STL**.
- **Documentación** de código.
- **Nombres completos y NIP de todos los autores al comienzo de cada fichero**.

ENTREGA DE LA PRÁCTICA

- Fecha límite: **01-12-2026** (incluido).
- Solo **entregará un miembro** del equipo.
- **practica4_NIP1_NIP2_NIP3.zip**.
- Dos ficheros ***"entrada.txt"*** y ***"salida.txt"***.
  - **Significativamente diferentes** a los propuestos.
- **Estructura plana**: **solo código fuente y los dos ficheros anteriores**.
- **Código ajeno**: solo el **TAD pila** y el **TAD fecha** suministrados, indicando su autoría.