ESTRATEGIAS DE PROGRAMACIÓN Y ESTRUCTURAS DE DATOS

Estructuras de Datos Básicas (III)

Árboles: Generales, Binarios y de Búsqueda Binaria

Equipo Docente de la Asignatura. Departamento LSI.

UNED

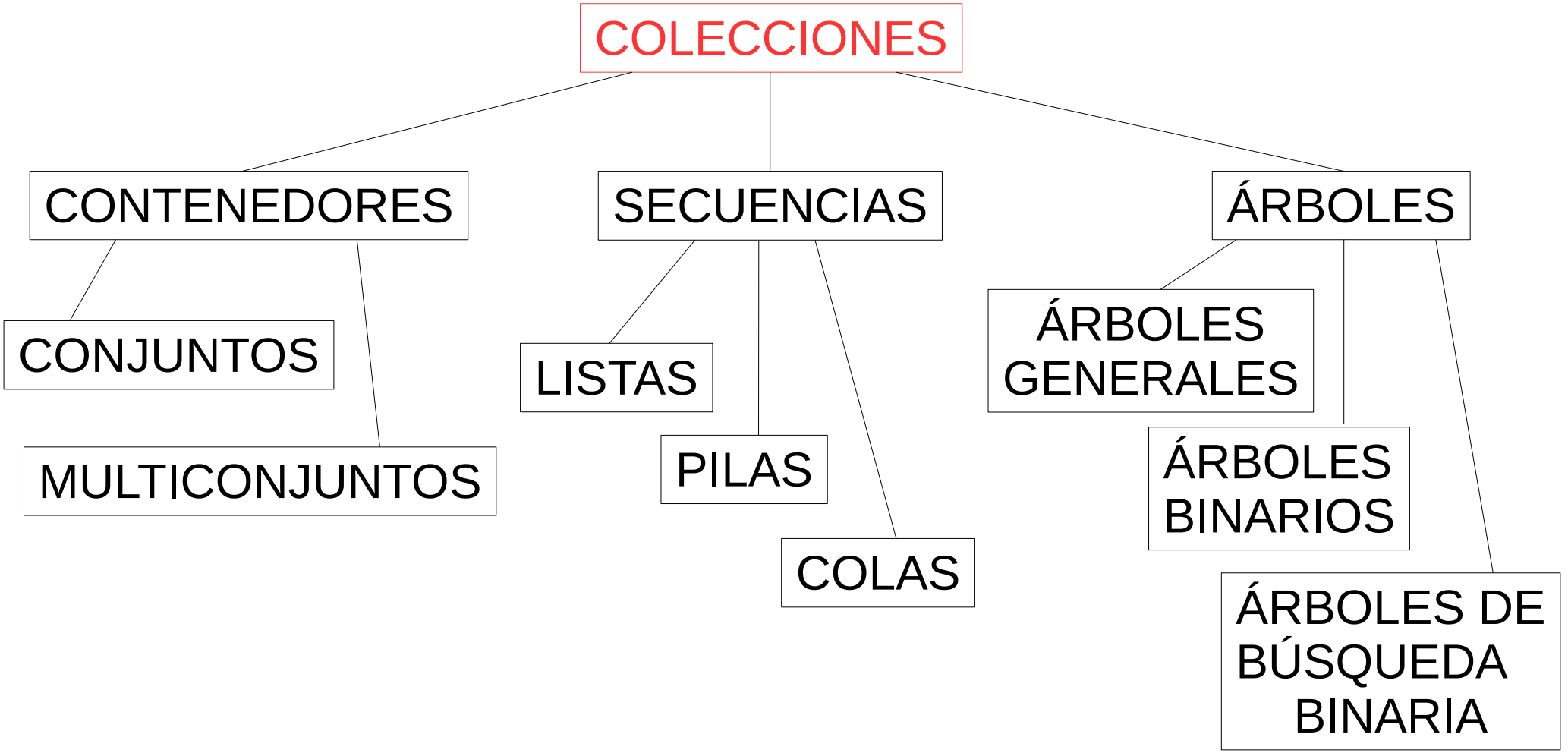
ETS de Ingeniería Informática

ÁRBOLES

Equipo Docente de la Asignatura. Departamento LSI.

UNED
ETS de Ingeniería Informática

Tipos Abstractos de Datos estudiados en esta asignatura

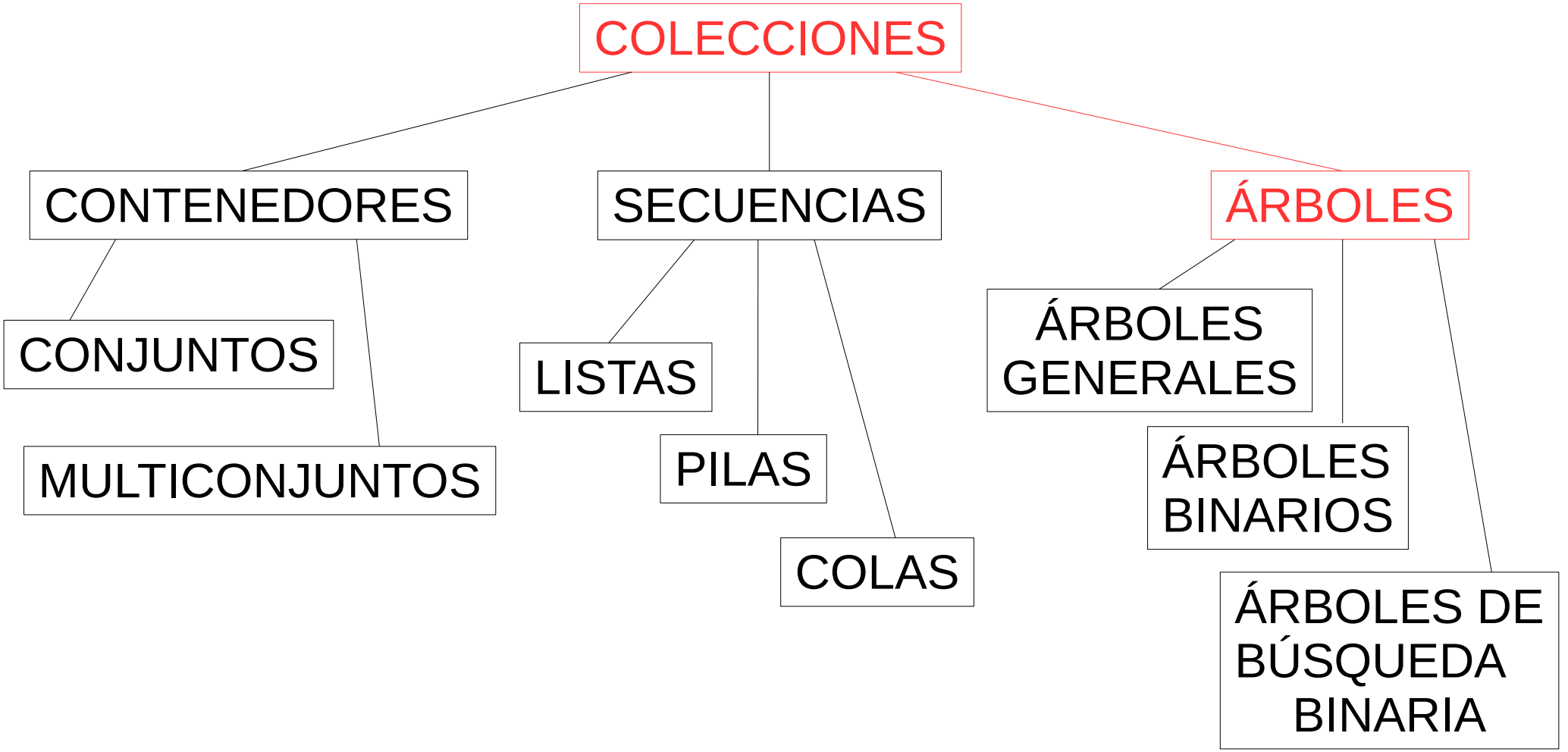


Equipo Docente de la Asignatura. Departamento LSI.

UNED

ETS de Ingeniería Informática

Tipos Abstractos de Datos estudiados en esta asignatura



Equipo Docente de la Asignatura. Departamento LSI.

UNED

ETS de Ingeniería Informática

Estructuras lineales vs. jerárquicas

- Estructuras lineales:
  - Todo elemento → un único predecesor (salvo el 1º)
  - Todo elemento → un único sucesor (salvo el último)
- Estructuras jerárquicas:
  - Cualquier elemento puede tener
    - varios predecesores
    - varios sucesores

Estructuras lineales vs. jerárquicas

- Estructuras jerárquicas, grandes familias:
  - Árboles (EPED) (caso particular de grafos)
  - Grafos (PREDA) (generalización de árboles)

ESTRUCTURAS DE DATOS
ÁRBOLES: DEFINICIONES

- Estructura jerárquica:
  - Un elemento puede preceder a ninguno, uno o más
    - **Padre** de un elemento: su predecesor
    - **Hijo** de un elemento: cualquiera de sus sucesores
  - Todo elemento es precedido por un único elemento
    - Salvo un elemento especial → **raíz** (sin padres)
  - Si un elemento no tiene hijos → **hoja**
  - Hijo de un árbol → árbol (**subárbol**)

Equipo Docente de la Asignatura. Departamento LSI.

UNED
ETS de Ingeniería Informática

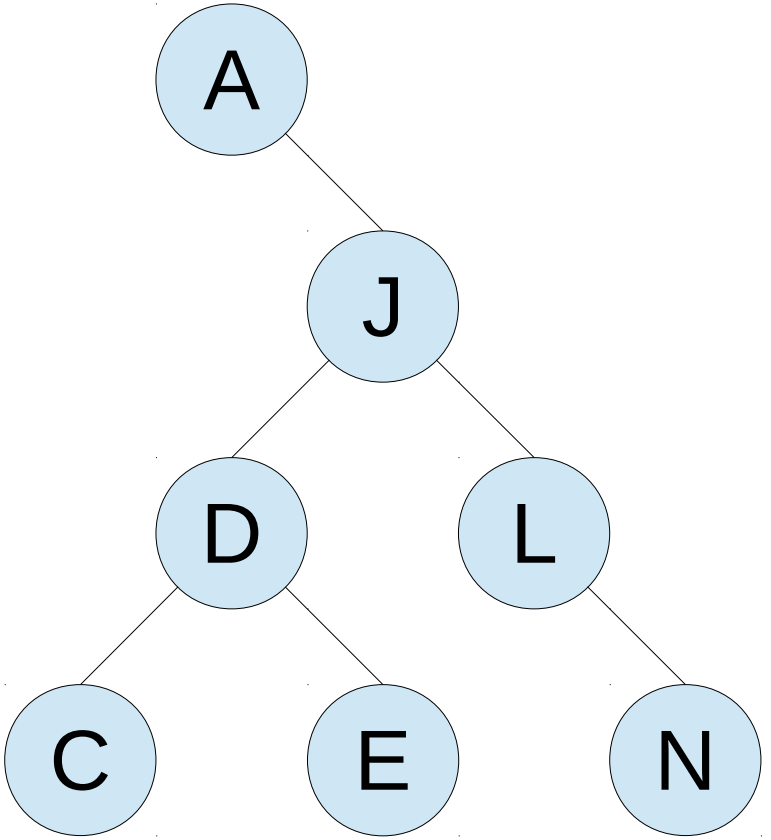
ESTRUCTURAS DE DATOS
ÁRBOLES: DEFINICIONES

- Anchura (**fan-out**): máximo número de hijos para un nodo
  - Árbol general: fan-out NO ACOTADO
  - Árbol binario: fan-out máximo 2
- **Distancia**: número mínimo de “saltos” de un elemento a otro
- **Altura de un elemento**: distancia de la raíz al elemento
- **Altura del árbol**: distancia máxima de la raíz a una hoja
- **Nivel**: conjunto de nodos de un árbol con la misma altura
  - ¿Cómo se llama una colección de árboles?
  - ¿Puede existir un árbol sin nodos?

Equipo Docente de la Asignatura. Departamento LSI.

UNED
ETS de Ingeniería Informática

ESTRUCTURAS DE DATOS
ÁRBOLES: DEFINICIONES



Equipo Docente de la Asignatura. Departamento LSI.

UNED
ETS de Ingeniería Informática

ESTRUCTURAS DE DATOS
ÁRBOLES

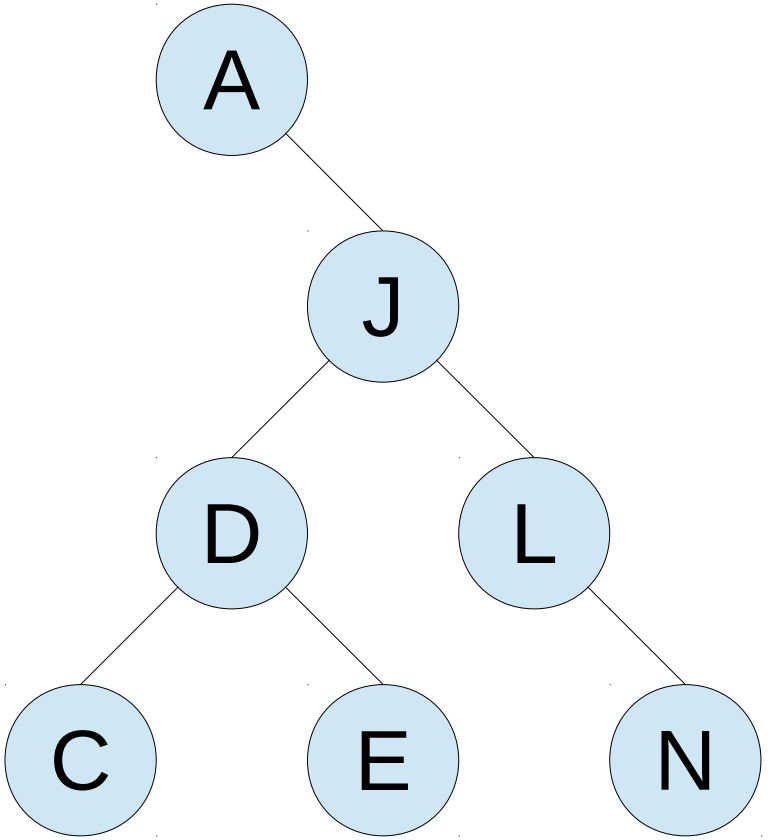
- ¿Qué operaciones hacen falta?
  - Obtener el elemento en la raíz del árbol
  - Obtener el número de hijos (árboles) del árbol
  - Comprobar si el árbol es una hoja
  - Obtener el fan-out del árbol
  - Obtener la altura del árbol
  - Obtener un iterador para el árbol según el recorrido elegido

ESTRUCTURAS DE DATOS
ÁRBOLES: RECORRIDO

- Árboles → varios hijos → varios recorridos
- **Recorridos en profundidad**:
  - Se baja al primer hijo y se recorre en profundidad
  - ...
  - Se baja al último hijo y se recorre en profundidad
  - ¿Cuándo se visita la raíz?
    - Antes de visitar los hijos → **preorden**
    - Después de visitar todos los hijos → **postorden**

ESTRUCTURAS DE DATOS
ÁRBOLES: RECORRIDO

- Recorridos en profundidad

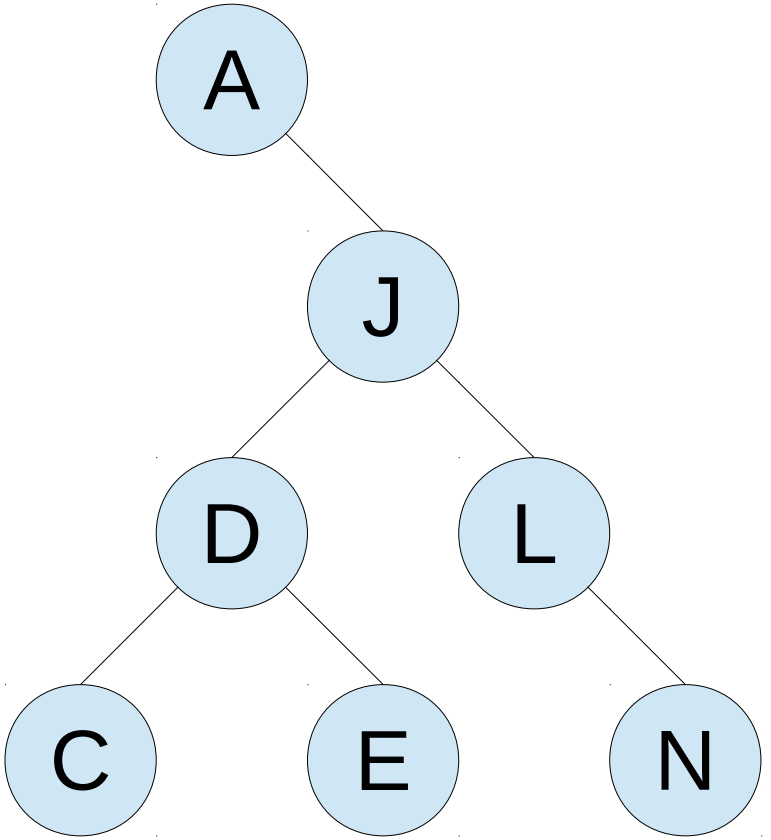


ESTRUCTURAS DE DATOS
ÁRBOLES: RECORRIDO

- **Recorridos en anchura**:
  - Se visita la raíz
  - Se visitan los hijos de la raíz (en orden)
  - Se visitan los hijos de los hijos de la raíz (en orden)
  - …
  - Se visita el último nivel del árbol
- Requieren una estructura adicional (cola)
- Consumen MUCHA memoria

ESTRUCTURAS DE DATOS
ÁRBOLES: RECORRIDO

- Recorridos en anchura



ESTRUCTURAS DE DATOS
ÁRBOLES

```
/* Representa un árbol, que es una colección cuyos elementos*
 * se organizan jerárquicamente.                             */

public abstract interface TreeIF<E> extends CollectionIF<E> {

  /* Obtiene el elemento situado en la raíz del árbol        *
   * @Pre: !isEmpty ();                                      *
   * @return el elemento que ocupa la raíz del árbol.       */
  public E getRoot ();

  /* Decide si el árbol es una hoja (no tiene hijos)         *
   * @return true sii el árbol es no vacío y no tiene hijos */
  public boolean isLeaf();
```

...

Equipo Docente de la Asignatura. Departamento LSI.

UNED ETS de Ingeniería Informática

ESTRUCTURAS DE DATOS
ÁRBOLES

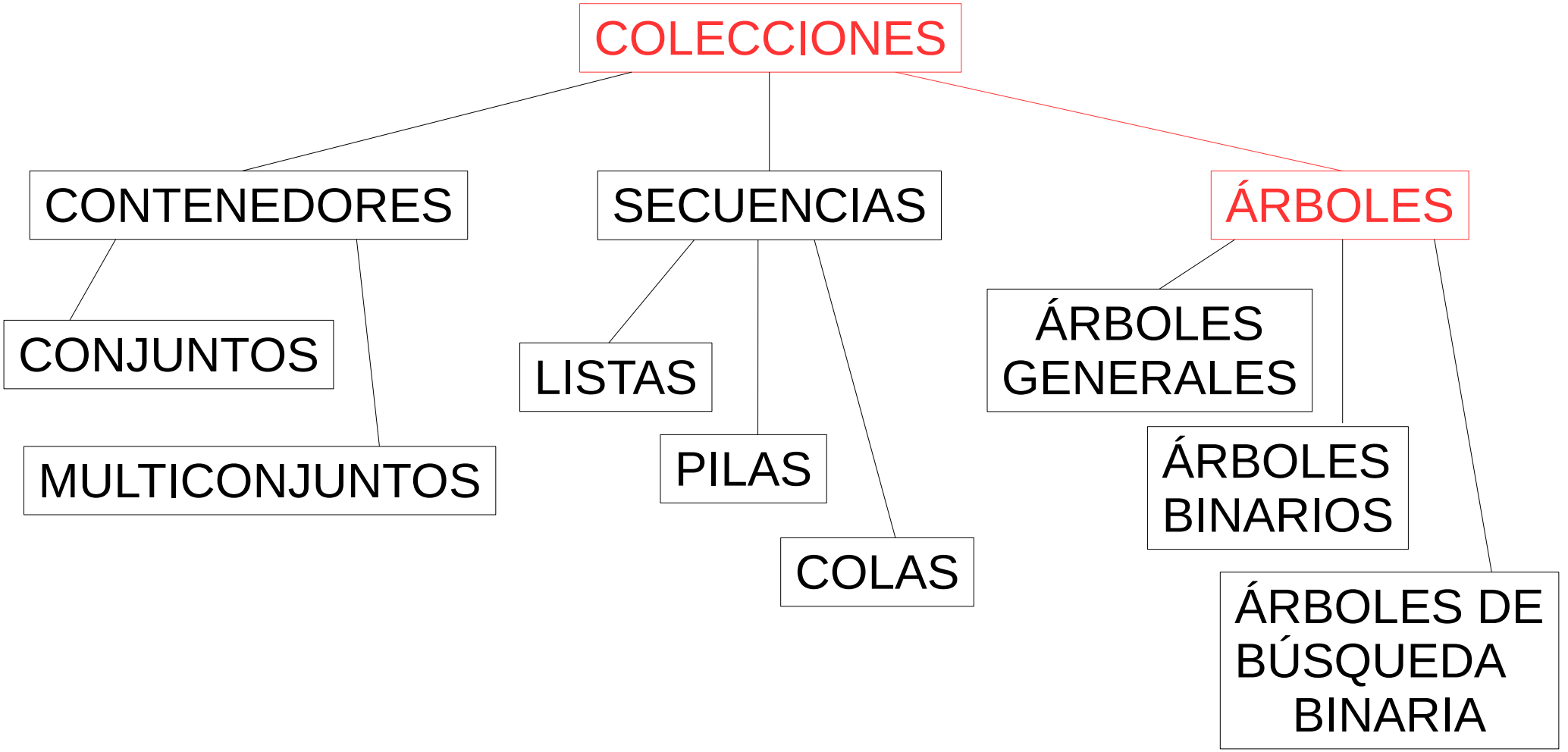
```
  /* Devuelve el número de hijos del árbol                          */
  public int getNumChildren ();

  /* Devuelve el fan-out del árbol: el número máximo de             *
   * hijos que tiene cualquier nodo del árbol                       */
  public int getFanOut ();

  /* Devuelve la altura del árbol: la distancia máxima desde*
   * la raíz a cualquiera de sus hojas                              */
  public int getHeight ();

  /* Obtiene un iterador para el árbol.                             *
   * @param mode el tipo de recorrido indicado por los             *
   * valores enumerados definidos en cada TAD concreto.            */
  public IteratorIF<E> iterator (Object mode);
}
```

ÁRBOLES:
ÁRBOLES GENERALES

Equipo Docente de la Asignatura. Departamento LSI.

UNED
ETS de Ingeniería Informática

Tipos Abstractos de Datos estudiados en esta asignatura

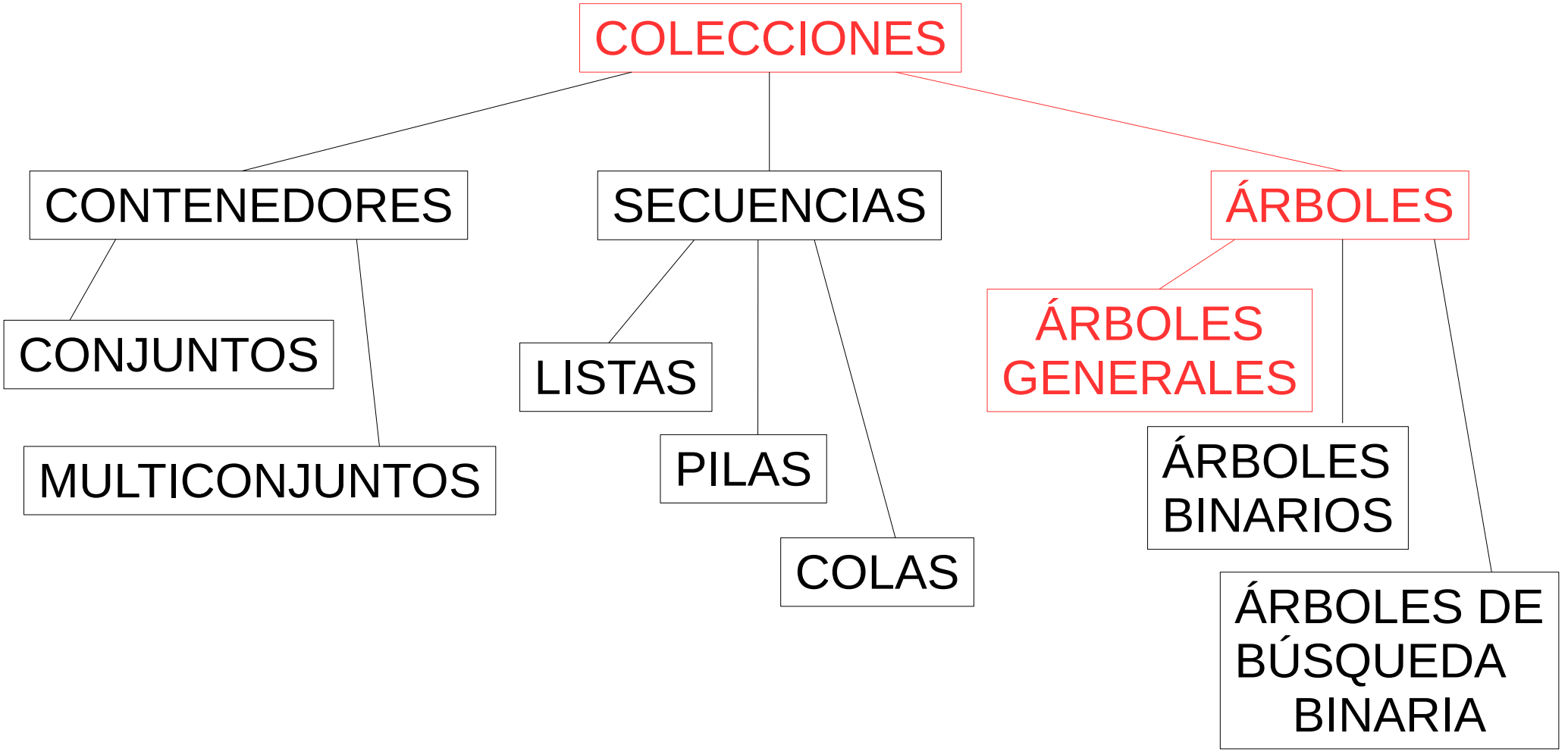


Equipo Docente de la Asignatura. Departamento LSI.

UNED

ETS de Ingeniería Informática

Tipos Abstractos de Datos estudiados en esta asignatura



Equipo Docente de la Asignatura. Departamento LSI.

UNED

ETS de Ingeniería Informática

ESTRUCTURAS DE DATOS
ÁRBOLES GENERALES

- Número de hijos no acotado (puede ser 0)
- ¿Qué operaciones hacen falta?
  - Modificar la raíz del árbol
  - Obtener todos los hijos
  - Obtener un hijo concreto
  - Insertar un hijo concreto
  - Eliminar un hijo concreto

ESTRUCTURAS DE DATOS
ÁRBOLES GENERALES

```
/* Representa un arbol general de elementos, en el que un   *
 * nodo puede tener cualquier número de hijos.              */
public interface GTreeIF<E> extends TreeIF<E> {
  /* Valor enumerado que indica los tipos de recorridos     *
   * ofrecidos por los árboles generales.                   */
  public enum IteratorModes {
    PREORDER, POSTORDER, BREADTH
  }

  /* Modifica la raíz del árbol.                            *
     * @param el elemento que se quiere poner como raíz del *
     * árbol.                                                */
  public void setRoot (E e);
```

...

Equipo Docente de la Asignatura. Departamento LSI.

UNED
ETS de Ingeniería Informática

ESTRUCTURAS DE DATOS
ÁRBOLES GENERALES

```
/* Obtiene los hijos del árbol llamante.                          *
 * @return la lista de hijos del árbol (en el orden en que *
 * estén almacenados en el mismo).                               */
public ListIF <GTreeIF<E>> getChildren ();

/* Obtiene el hijo que ocupa la posición dada por parámetro*
 * @param pos la posición del hijo que se desea obtener,   *
 *  comenzando en 1.                                              *
 * @Pre 1 <= pos <= getChildren().size ();                        *
 * @return el árbol hijo que ocupa la posición pos.              */
public GTreeIF<E> getChild (int pos);
```

...

Equipo Docente de la Asignatura. Departamento LSI.

UNED
ETS de Ingeniería Informática

ESTRUCTURAS DE DATOS
ÁRBOLES GENERALES

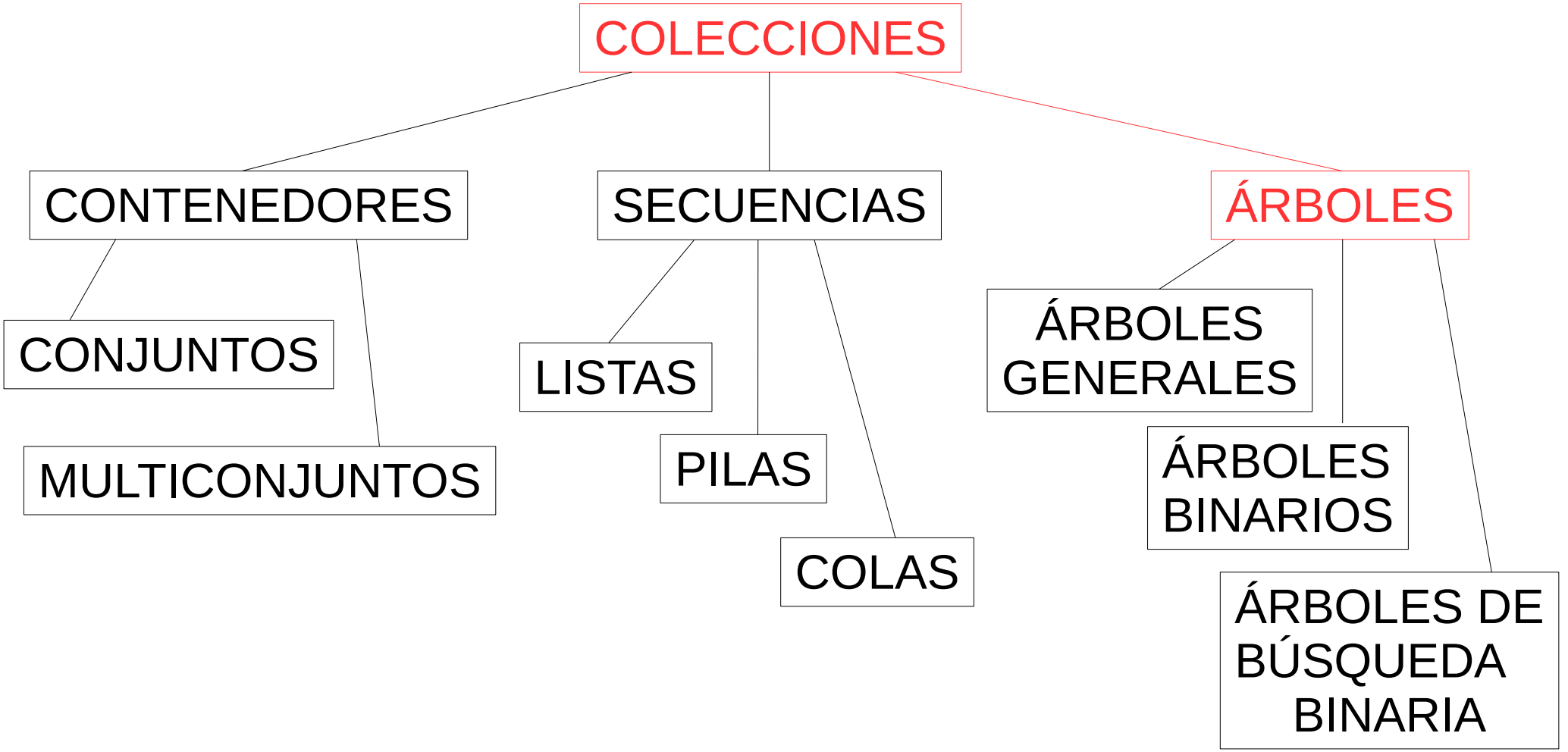
```
  /* Inserta un árbol como hijo en la posición pos.              *
   * @param pos la posición que ocupará el árbol entre sus      *
   * hermanos, comenzando en 1.                                 *
   * Si pos == getChildren().size () + 1, se añade como         *
   * último hijo.                                               *
   * @param e el hijo que se desea insertar.                    *
   * @Pre 1<= pos <= getChildren ().size () + 1                 */
  public void addChild (int pos, GTreeIF<E> e);

  /* Elimina el hijo que ocupa la posición parámetro.           *
   * @param pos la posición del hijo con base 1.                *
   * @Pre 1 <= pos <= getChildren ().size ();                   */
  public void removeChild (int pos);
}
```

ÁRBOLES:
ÁRBOLES BINARIOS

Equipo Docente de la Asignatura. Departamento LSI.

UNED
ETS de Ingeniería Informática

Tipos Abstractos de Datos estudiados en esta asignatura

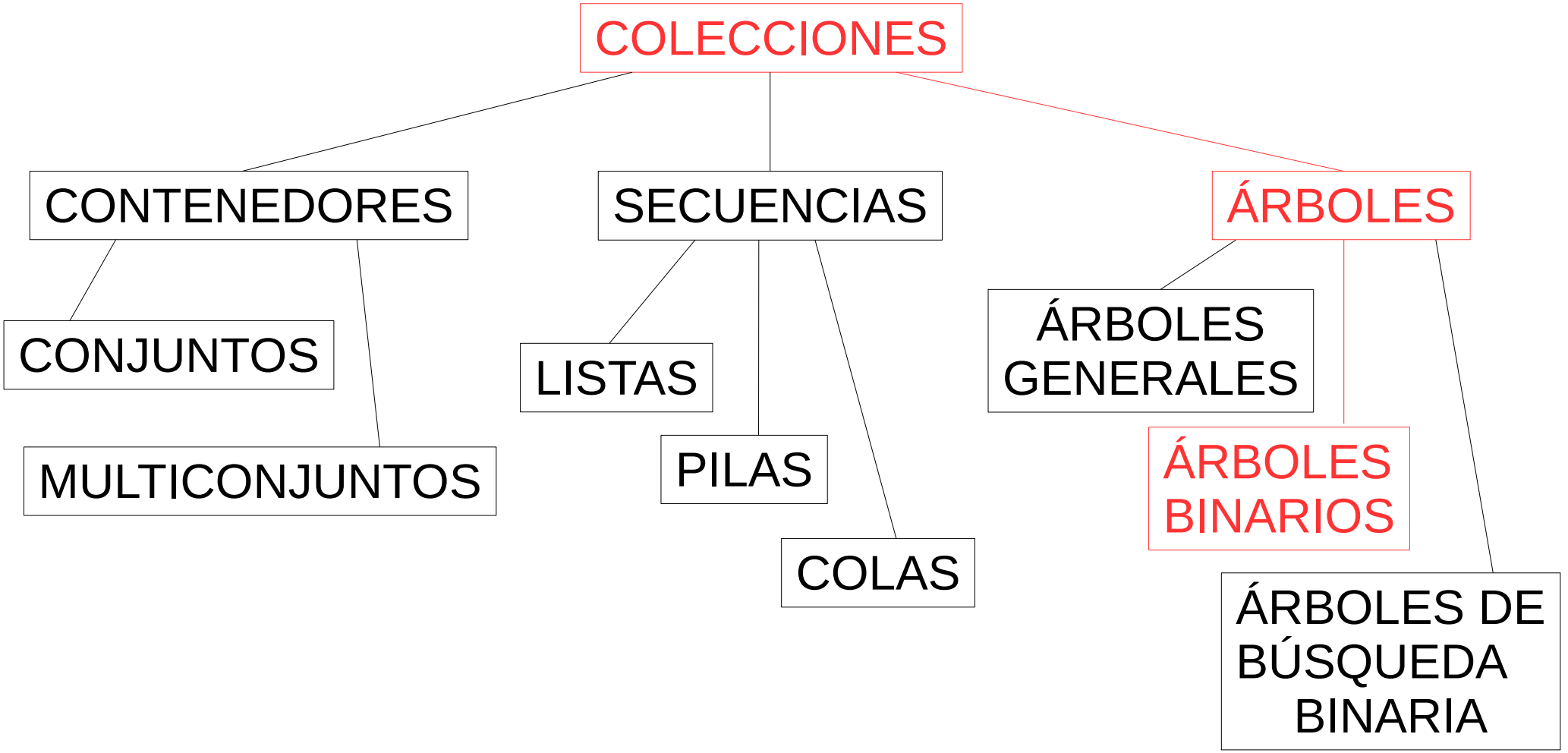


Equipo Docente de la Asignatura. Departamento LSI.

UNED

ETS de Ingeniería Informática

Tipos Abstractos de Datos estudiados en esta asignatura



Equipo Docente de la Asignatura. Departamento LSI.

UNED

ETS de Ingeniería Informática

ESTRUCTURAS DE DATOS
ÁRBOLES BINARIOS

- Número de hijos acotado: máximo 2
- Normalmente: “hijo izquierdo” e “hijo derecho”
- ¿Qué operaciones hacen falta?
  - Modificar la raíz
  - Acceder, modificar y eliminar ambos hijos

ESTRUCTURAS DE DATOS
ÁRBOLES BINARIOS: RECORRIDO

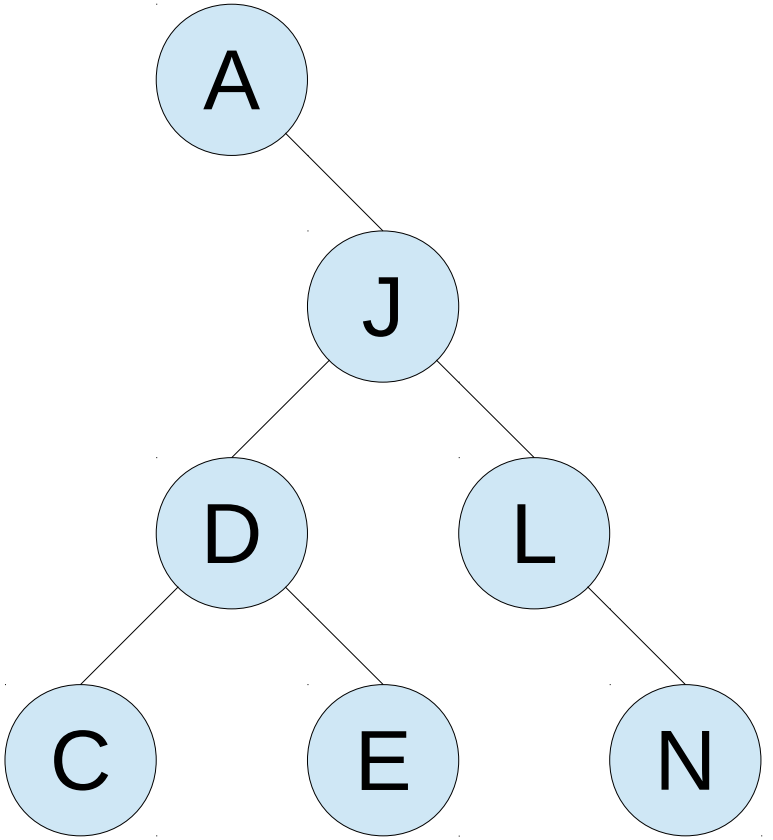
- **Recorridos en profundidad**:
  - ¿Cuándo se visita la raíz?
    - Antes de visitar los hijos → **preorden**
    - Después de visitar todos los hijos → **postorden**
    - Tras visitar H.I. y antes de visitar H.D. → **inorden**
- **Recorridos en anchura**
  - Se recorren los nodos de izquierda a derecha
  - Se recorren los nodos de derecha a izquierda

Equipo Docente de la Asignatura. Departamento LSI.

UNED
ETS de Ingeniería Informática

ESTRUCTURAS DE DATOS
ÁRBOLES BINARIOS: RECORRIDO

- Recorrido en inorden

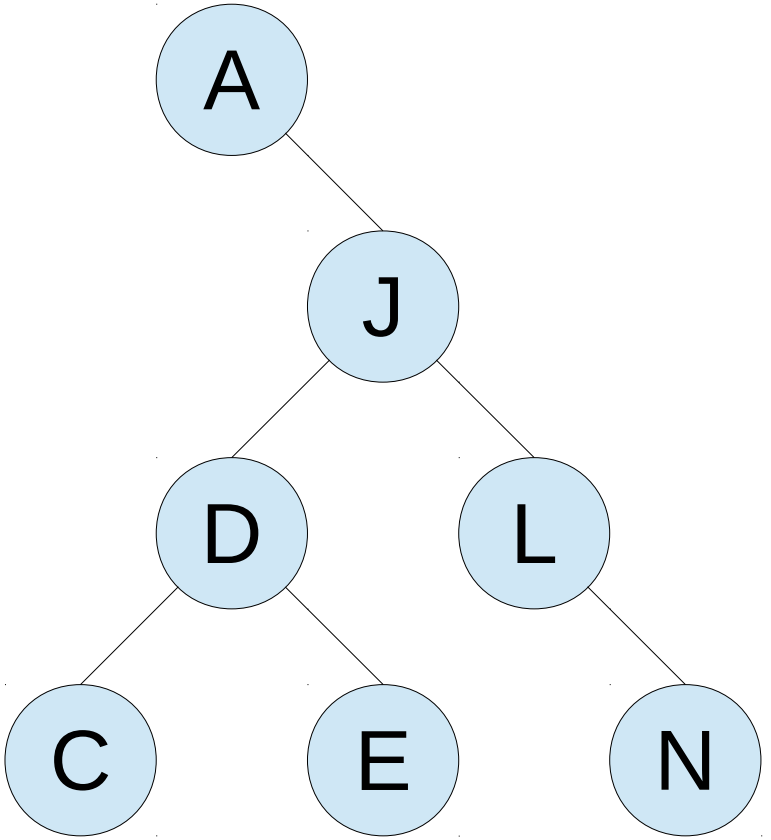


Equipo Docente de la Asignatura. Departamento LSI.

UNED
ETS de Ingeniería Informática

ESTRUCTURAS DE DATOS
ÁRBOLES BINARIOS: RECORRIDO

- Recorrido en anchura de derecha a izquierda



ESTRUCTURAS DE DATOS
ÁRBOLES BINARIOS

```
/* Representa un arbol binario de elementos, en el que un    *
 * nodo puede tener, a lo sumo, dos hijos (fan-out <= 2 para *
 * todos los nodos).                                         */
public interface BTreeIF<E> extends TreeIF<E>{
  /* Valor enumerado que indica los tipos de recorrido      *
   * ofrecidos por los árboles de binarios.                 */
  public enum IteratorModes {
    PREORDER, POSTORDER, BREADTH, INORDER, RLBREADTH
  }

  /* Modifica la raíz del árbol.                            *
   * @param el elemento que se quiere poner como raíz del   *
   * árbol.                                                 */
  public void setRoot (E e);
```

...

Equipo Docente de la Asignatura. Departamento LSI.

UNED
ETS de Ingeniería Informática

ESTRUCTURAS DE DATOS
ÁRBOLES BINARIOS

```
/* Obtiene el hijo izquierdo del árbol llamante.              *
 * @return el hijo izquierdo del árbol llamante.              */
public BTreeIF<E> getLeftChild ();

/* Pone el árbol parámetro como hijo izquierdo del árbol     *
 * llamante. Si ya había hijo izquierdo, el antiguo dejará   *
 * de ser accesible (se pierde).                             *
 * @Pre: !isEmpty()                                          *
 * @param child el árbol que se debe poner como hijo         *
 *        izquierdo.                                          */
public void setLeftChild (BTreeIF <E> child);

/* Elimina el hijo izquierdo del árbol.                      */
public void removeLeftChild ();
                                    ...
```

Equipo Docente de la Asignatura. Departamento LSI.

UNED
ETS de Ingeniería Informática

ESTRUCTURAS DE DATOS
ÁRBOLES BINARIOS

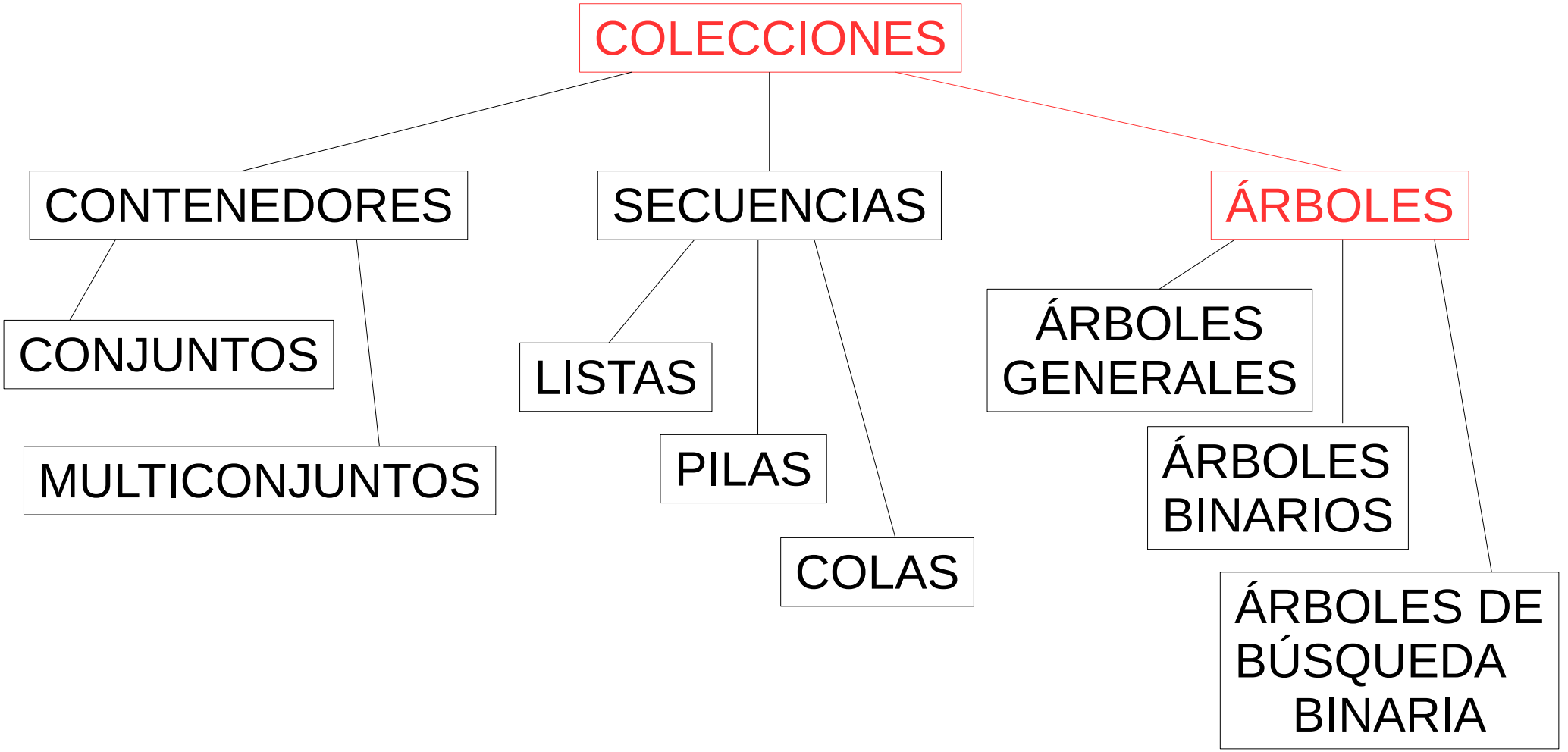
```
  /* Obtiene el hijo derecho del árbol llamante.                    *
   * @return el hijo derecho del árbol llamante                     */
  public BTreeIF<E> getRightChild ();

  /* Pone el árbol parámetro como hijo derecho del árbol            *
   * llamante. Si ya había hijo izquierdo, el antiguo dejará        *
   * de ser accesible (se pierde).                                  *
   * @Pre: !isEmpty()                                               *
   * @param child el árbol que se debe poner como hijo              *
   *        derecho.                                                */
  public void setRightChild (BTreeIF <E> child);

  /* Elimina el hijo derecho del árbol.                             */
  public void removeRightChild ();
}
```

ÁRBOLES:
ÁRBOLES DE BÚSQUEDA BINARIA

Equipo Docente de la Asignatura. Departamento LSI.

UNED
ETS de Ingeniería Informática

Tipos Abstractos de Datos estudiados en esta asignatura

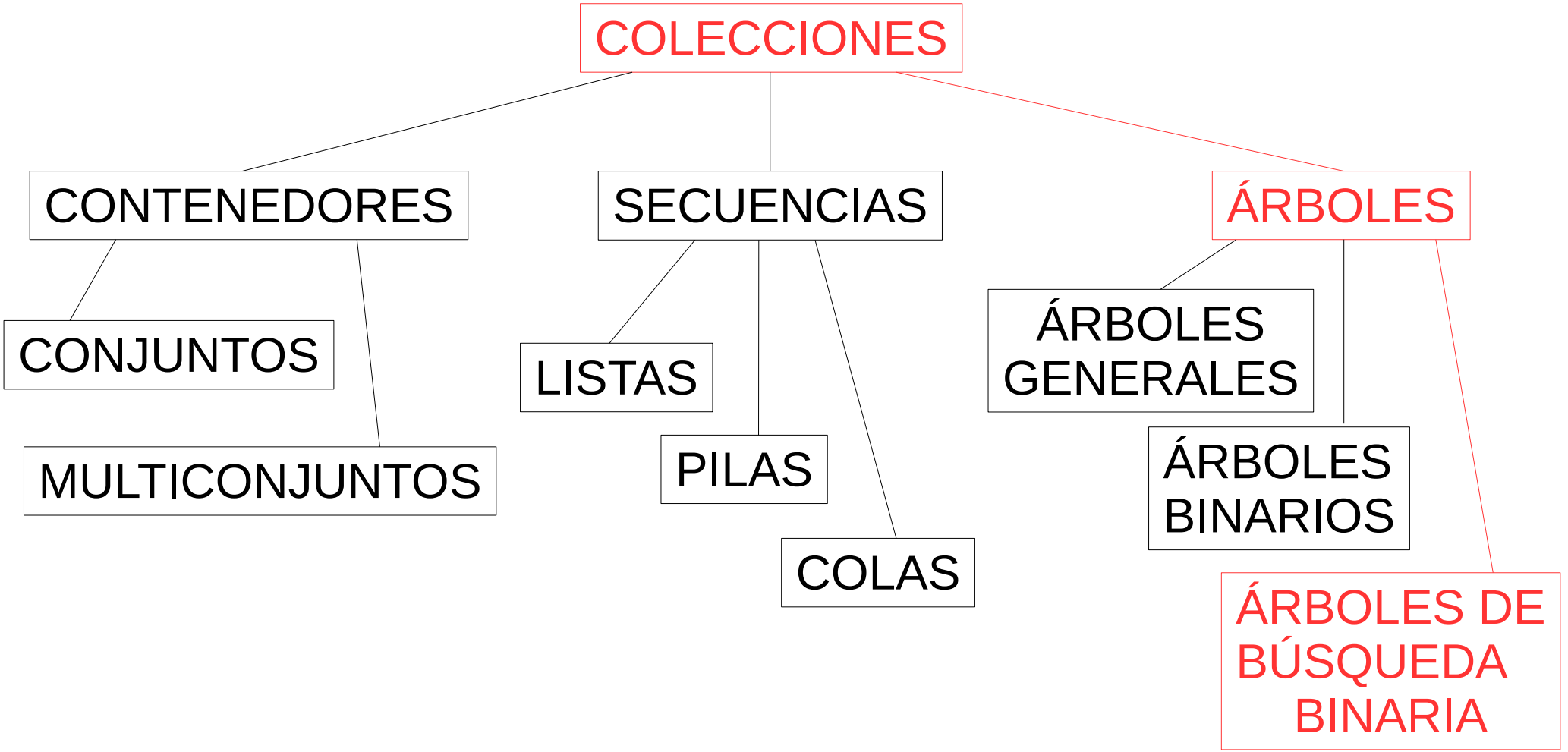


Equipo Docente de la Asignatura. Departamento LSI.

UNED

ETS de Ingeniería Informática

Tipos Abstractos de Datos estudiados en esta asignatura



Equipo Docente de la Asignatura. Departamento LSI.

UNED

ETS de Ingeniería Informática

ESTRUCTURAS DE DATOS
ÁRBOLES DE BÚSQUEDA BINARIA

- Árboles binarios especializados para búsquedas
- Búsqueda binaria → datos ORDENADOS en el árbol
  - Todos los mayores que la raíz en un hijo
  - Todos los menores que la raíz en el otro hijo
- Buscar:
  - Igual que la raíz → encontrado
  - Mayor que la raíz → buscar en hijo de elementos mayores
  - Menor que la raíz → buscar en hijo de elementos menores

Equipo Docente de la Asignatura. Departamento LSI.

UNED ETS de Ingeniería Informática

ESTRUCTURAS DE DATOS
ÁRBOLES DE BÚSQUEDA BINARIA

- Si modificamos el árbol “desde fuera”:
  - Podemos perder la propiedad de orden
  - Tendríamos que asegurar que se cumple
- Solución:
  - No permitir modificar el árbol “desde fuera”
- Operaciones permitidas:
  - Añadir y eliminar elementos
  - Consultar el orden de almacenamiento

ESTRUCTURAS DE DATOS
ÁRBOLES DE BÚSQUEDA BINARIA

```
/* Representa un árbol de búsqueda binaria, en el que los    *
 * elementos se organizan automáticamente según su orden.    */
public interface BSTreeIF<E extends Comparable<E>> extends
TreeIF<E> {
  /* Valor enumerado que indica los tipos de recorrido       *
   * ofrecidos por los árboles de búsqueda binaria.          */
  public enum IteratorModes {
    DIRECTORDER, REVERSEORDER
  }

  /* Valor enumerado que indica cuál es la ordenación de los *
   * elementos dentro del árbol (ascendente o descendente).  */
  public enum Order {
    ASCENDING, DESCENDING
  }
                                        ...
```

Equipo Docente de la Asignatura. Departamento LSI.

UNED ETS de Ingeniería Informática

ESTRUCTURAS DE DATOS
ÁRBOLES DE BÚSQUEDA BINARIA

```
  /* Añade un elemento no contenido previamente en el árbol  *
   * @Pre: !contains(e)                                      *
   * @Post: contains(e)                                      */
  public void add(E e);

  /* Elimina un elemento previamente contenido en el árbol   *
   * @Pre: contains(e)                                       *
   * @Post: !contains(e)                                     */
  public void remove(E e);

  /* Devuelve el orden de almacenamiento de los elementos en *
   * el árbol                                                */
  public Order getOrder();
}
```

Equipo Docente de la Asignatura. Departamento LSI.

UNED
ETS de Ingeniería Informática

ESTRATEGIAS DE PROGRAMACIÓN Y ESTRUCTURAS DE DATOS

Estructuras de Datos Básicas (III)

Árboles: Generales, Binarios y de Búsqueda Binaria

Equipo Docente de la Asignatura. Departamento LSI.

UNED

ETS de Ingeniería Informática