ESTRATEGIAS DE PROGRAMACIÓN Y ESTRUCTURAS DE DATOS

Estructuras de Datos Básicas (I)

Interfaces, Iteradores, Colecciones y Contenedores

Equipo Docente de la Asignatura. Departamento LSI.

UNED ETS de Ingeniería Informática

INTERFACES

Equipo Docente de la Asignatura. Departamento LSI.

UNED
ETS de Ingeniería Informática

INTERFACES

- Interfaces Java → representa un TAD
  - Define perfil operaciones → perfil de los métodos
  - Satisfacer Interfaz → implementar TODOS sus métodos
- ¿Cómo definir propiedades? → Anotaciones:
  - @param → descripción parámetros
  - @return → descripción valor devuelto (si procede)
  - @Pre → precondición
  - @Post → postcondición

INTERFACES

- Precondición:
  - ¿Qué espera el método a su entrada?
  - El método NO LAS COMPRUEBA
  - ¿Quién lo hace? → quien llama al método
- Postcondición:
  - ¿Qué se compromete a cumplir el método a la salida?
  - Relación entre entrada y salida

INTERFACES

- Definiremos unos TAD en función de otros
  - Relación de extensión
  - Lista → extiende Secuencia → extiende Colección
- Cada TAD nombres de operaciones diferentes
  - Almacenar → distinto nombre en distintos TAD
  - Lista → insertar ; Pila → apilar
- Interfaz → NO EJECUTABLE
  - Requiere programar clases que implementen la interfaz

ITERADORES

Equipo Docente de la Asignatura. Departamento LSI.

UNED
ETS de Ingeniería Informática

ITERADORES

- Interfaz para recorrer un TAD
  - Sencillos de usar
  - Eficientes (generados por el TD)
  - Costosos de producir
  - Pueden violar la semántica del TAD
- Operaciones:

ITERADORES

- Interfaz para recorrer un TAD
  - Sencillos de usar
  - Eficientes (generados por el TD)
  - Costosos de producir
  - Pueden violar la semántica del TAD
- Operaciones:
  - ¿Hay más elementos por recorrer?
  - Dame el siguiente elemento
  - Vuelve a situarte al principio

ITERADORES

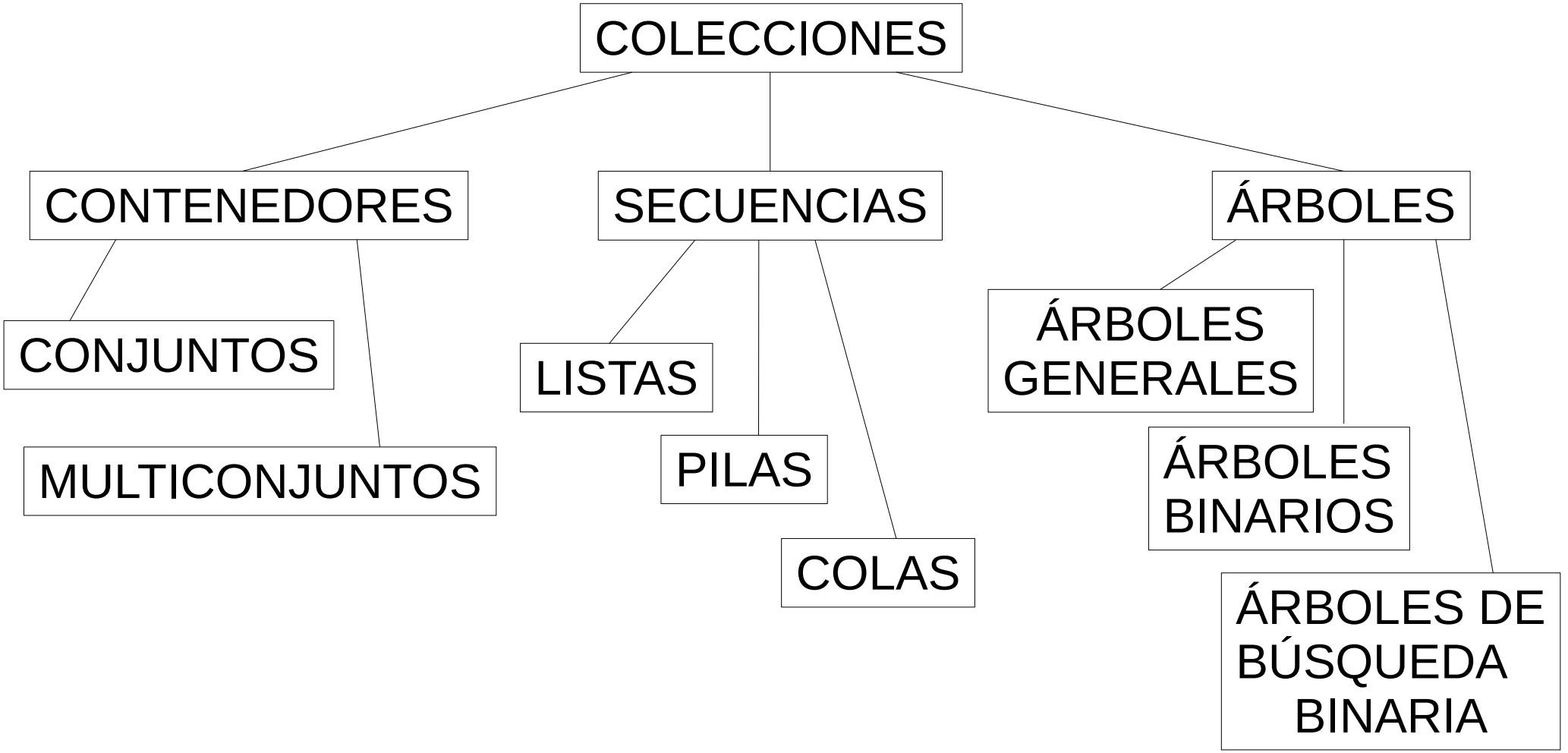
```
/* Representa un iterador de elementos.                          */
public interface IteratorIF<E> {

  /* Obtiene el siguiente elemento de la iteración.              *
   * @Pre:  hasNext()                                            *
   * @return: el siguiente elemento de la iteración.             */
  public E getNext ();

 /* Comprueba si aún quedan elementos por iterar.                *
  * @return true sii el iterador dispone de más elementos.      */
  public boolean hasNext ();

 /* Vuelve la posición del iterador al principio. Esto           *
  * permite reutilizar un iterador sin crear otro nuevo.        */
  public void reset ();
}
```

COLECCIONES

Equipo Docente de la Asignatura. Departamento LSI.

UNED
ETS de Ingeniería Informática

Tipos Abstractos de Datos estudiados en esta asignatura

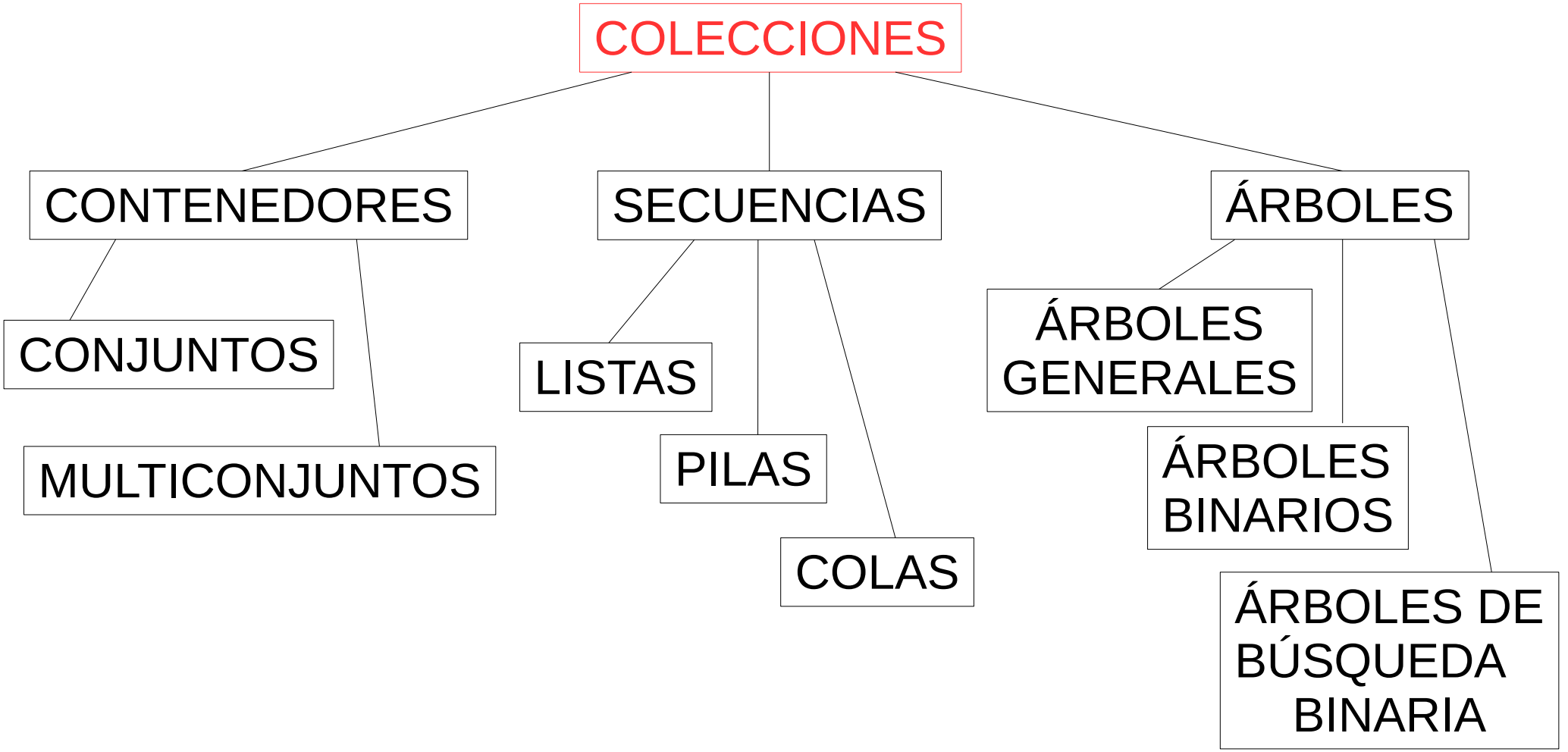


Equipo Docente de la Asignatura. Departamento LSI.

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 COLECCIONES

- Sin restricciones adicionales: sólo importa si un elemento está o no
  - TODOS los demás TAD extienden Colección
- ¿Qué operaciones hacen falta?

ESTRUCTURAS DE DATOS COLECCIONES

- Sin restricciones adicionales: sólo importa si un elemento está o no
  - TODOS los demás TAD extienden Colección
- ¿Qué operaciones hacen falta?
  - Añadir, eliminar, obtener y modificar elementos
  - Tamaño
  - Vaciar, ¿está vacía?
  - ¿Pertenece un elemento?

ESTRUCTURAS DE DATOS
COLECCIONES

- Sin restricciones adicionales: sólo importa si un elemento está o no
  - TODOS los demás TAD extienden Colección
- ¿Qué operaciones hacen falta?
  - ~~Añadir, eliminar, obtener y modificar elementos~~
  - Tamaño
  - Vaciar, ¿está vacía?
  - ¿Pertenece un elemento?

ESTRUCTURAS DE DATOS
COLECCIONES

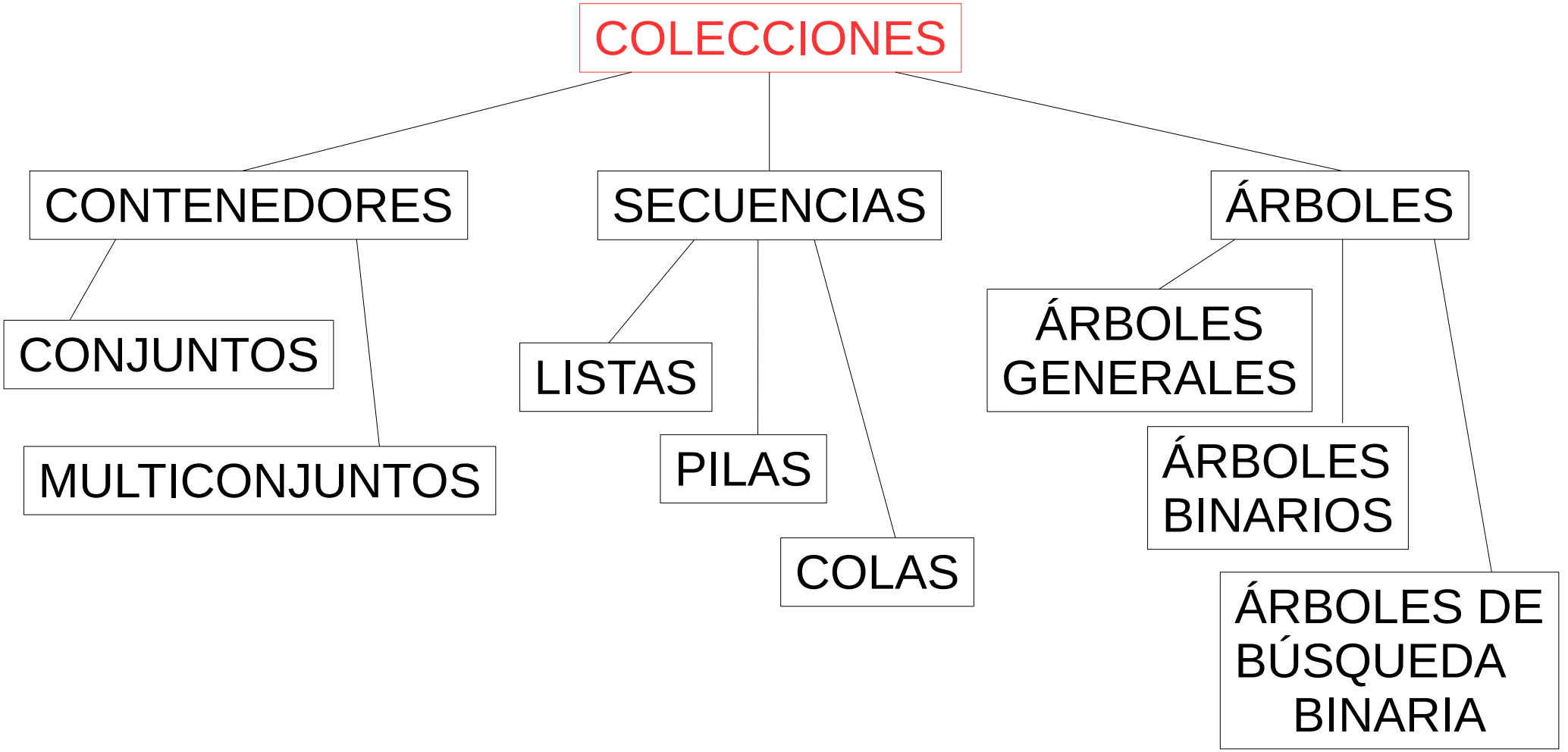
```
/* Representa una colección de elementos, sin ningún tipo de        *
 *  relación entre ellos más que la pertenencia a la misma          *
 *  colección.                                                      */
public interface CollectionIF<E> {

  /* Devuelve el número de elementos de la colección.              */
  public int size ();

  /* Devuelve true sii la colección no contiene elementos.         */
  public boolean isEmpty ();

  /* Devuelve true sii e está en la colección.                     */
  public boolean contains (E e);

  /* Elimina todos los elementos de la colección.                  */
  public void clear ();
}
```

CONTENEDORES

Equipo Docente de la Asignatura. Departamento LSI.

UNED
ETS de Ingeniería Informática

Tipos Abstractos de Datos estudiados en esta asignatura

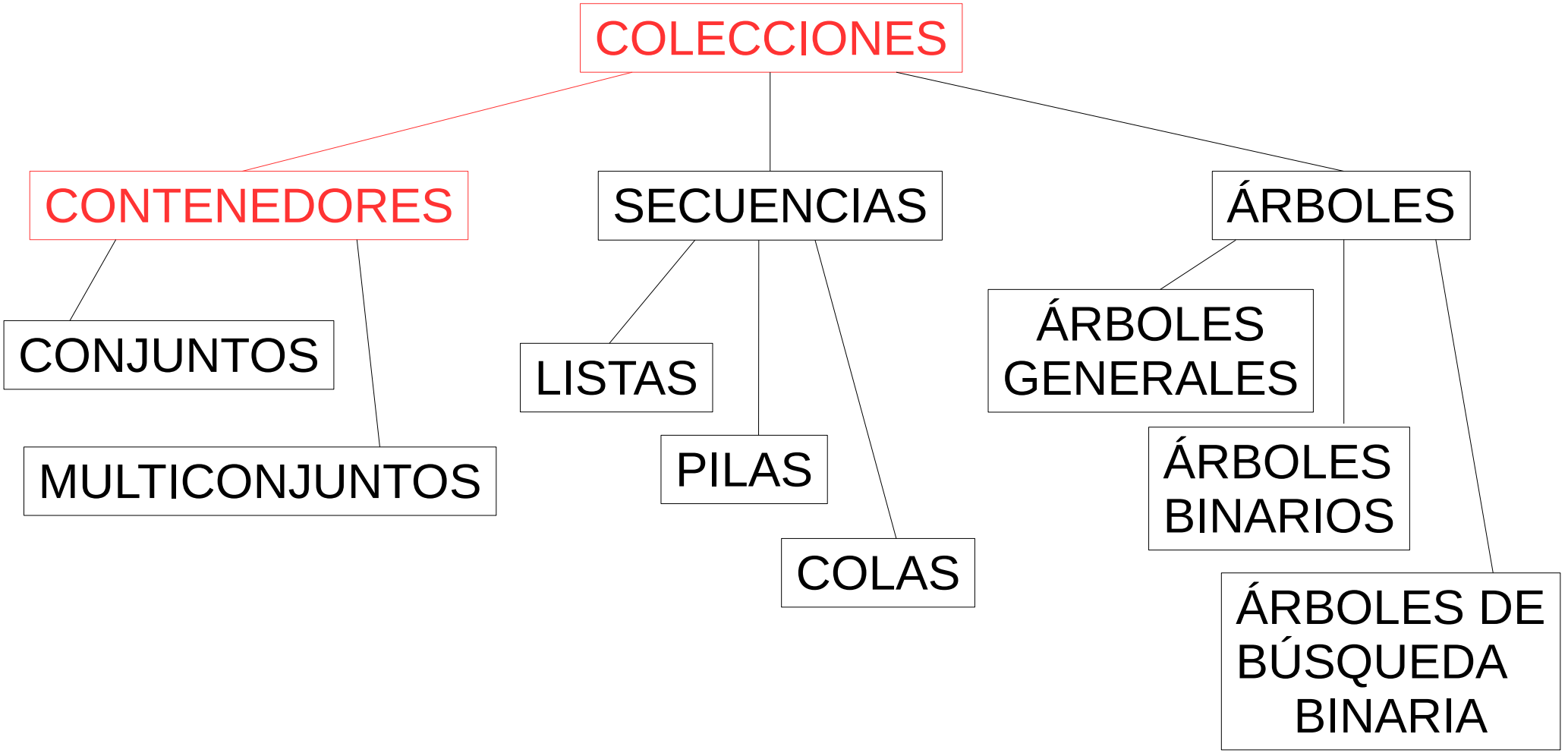


Equipo Docente de la Asignatura. Departamento LSI.

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 CONTENEDORES

- Colecciones de elementos sin orden entre sí
  - Importa si un elemento está o no está
  - Se pueden añadir y eliminar elementos
- ¿Qué operaciones hacen falta?
  - Añadir un elemento
  - Eliminar un elemento existente
  - Recorrer los elementos

ESTRUCTURAS DE DATOS CONTENEDORES

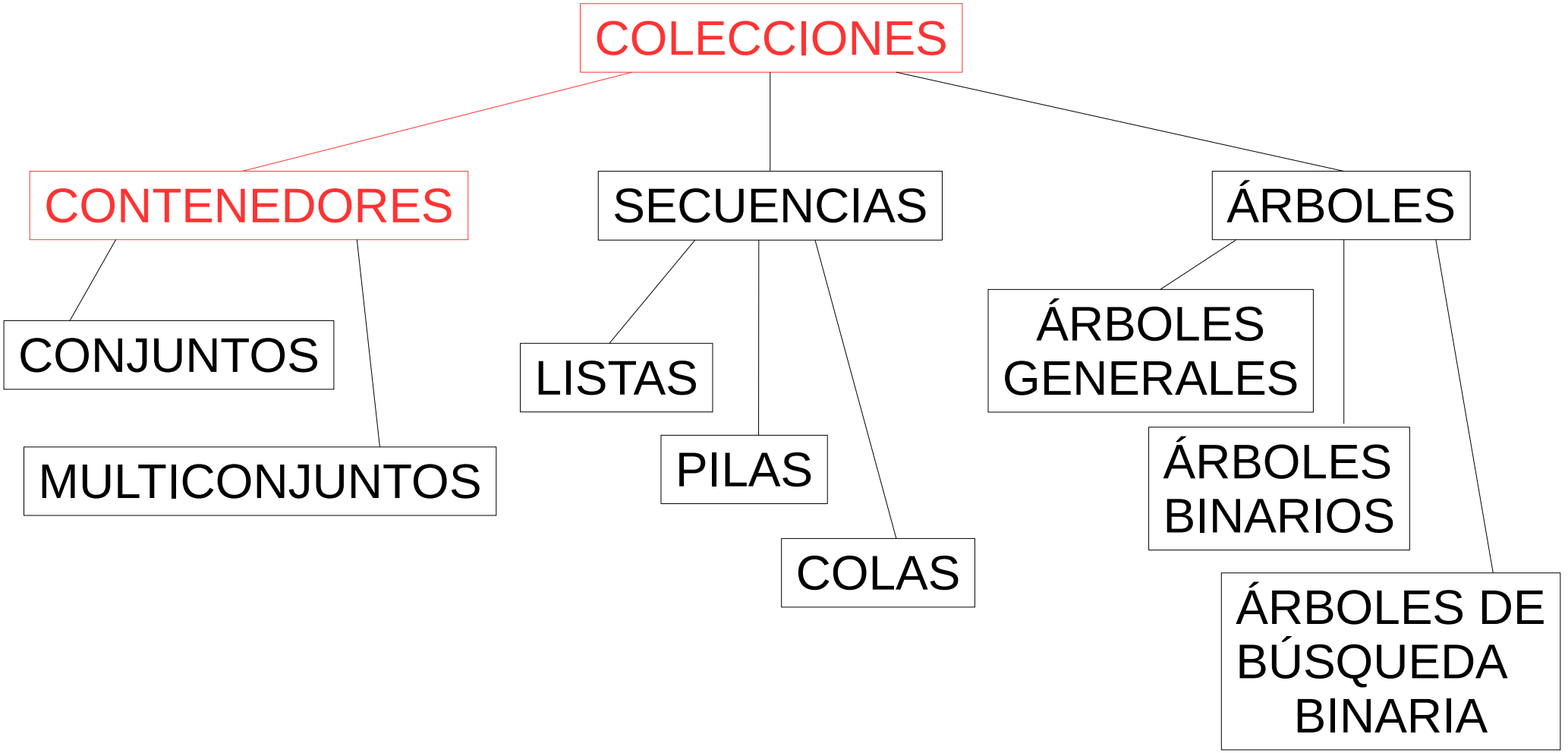
```
/* Representa un contenedor, que es una colección de          *
 * elementos que no guardan ningún orden entre sí.            */
public interface ContainerIF<E> extends CollectionIF<E> {

  /* Añade un elemento al contenedor                          */
  public void add (E e);

  /* Elimina un elemento e del contenedor                     *
   * @pre:  this.contains(e)                                  *
   * @post: !this.contains(e)                                 */
  public void remove (E e);

  /* Devuelve un iterador para el contenedor                  */
  public IteratorIF<E> iterator ();
}
```

CONTENEDORES: CONJUNTOS

Equipo Docente de la Asignatura. Departamento LSI.

UNED
ETS de Ingeniería Informática

Tipos Abstractos de Datos estudiados en esta asignatura

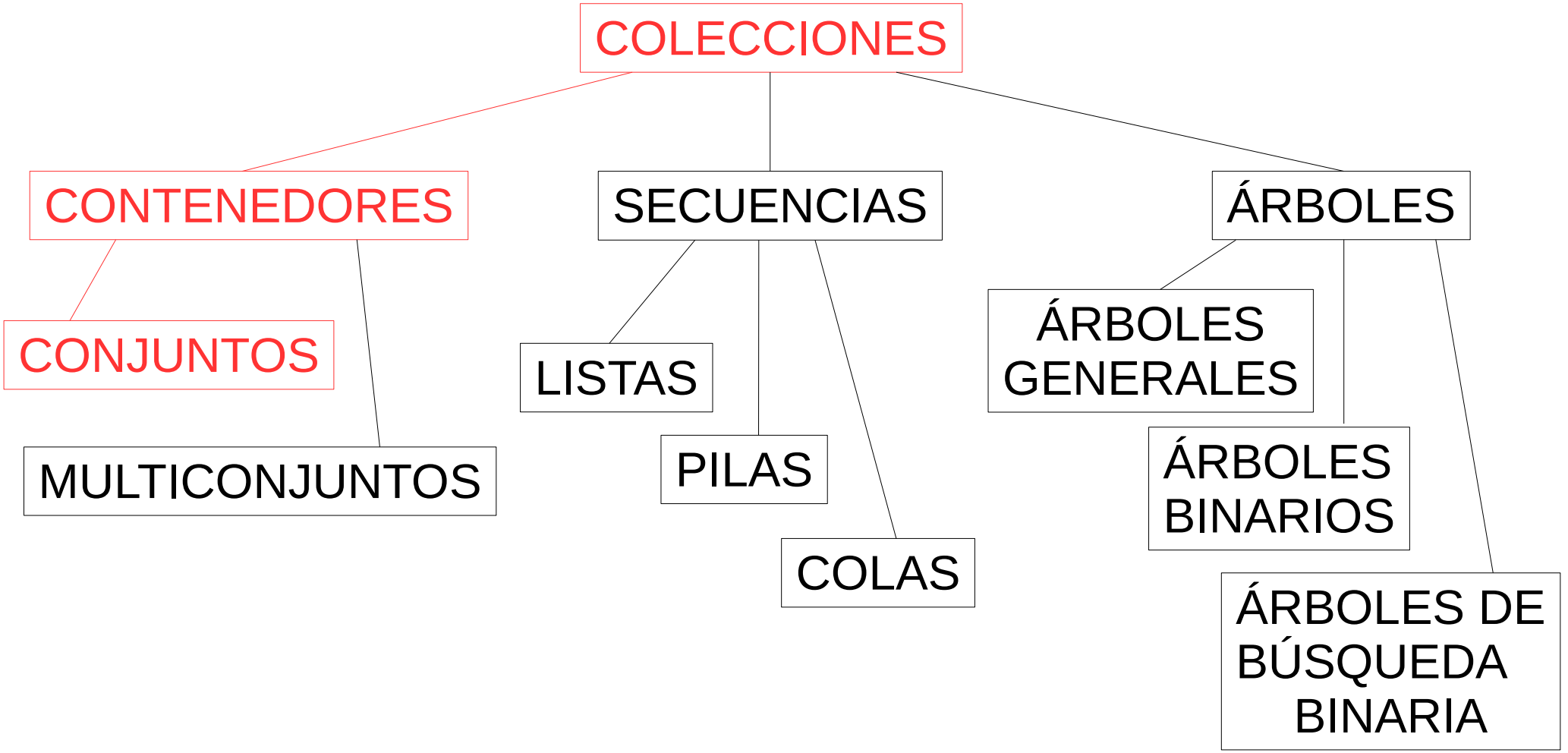


Equipo Docente de la Asignatura. Departamento LSI.

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 CONJUNTOS

- Concepto matemático de conjunto finito
  - Cada elemento está SÓLO una vez
- ¿Qué operaciones hacen falta?
  - Operaciones conjuntistas: unión, intersección, diferencia
  - Comprobar si es subconjunto de otro

ESTRUCTURAS DE DATOS
CONJUNTOS

```
/* Representa un conjunto, que es un contenedor que permite *
 * almacenar elementos que serán únicos dentro del conjunto */
public interface SetIF<E> extends ContainerIF<E> {

  /* Realiza la unión del conjunto llamante con el parámetro*/
  public void union (SetIF<E> s);

  /* Realiza la intersección del conjunto llamante con el   *
   * parámetro                                              */
  public void intersection (SetIF<E> s);
```

...

ESTRUCTURAS DE DATOS
CONJUNTOS

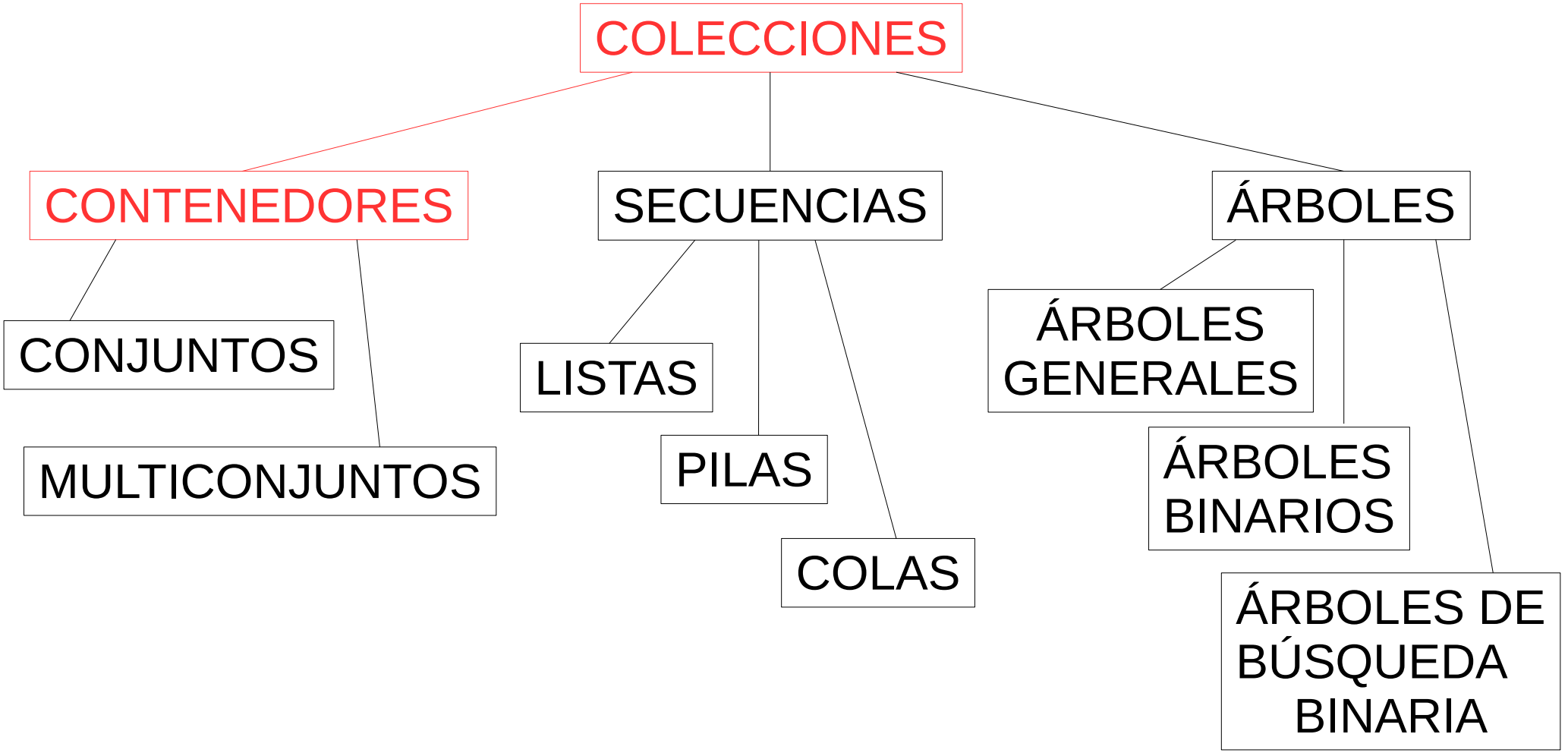
```
  /* Realiza la diferencia del conjunto llamante con el      *
   * parámetro (los elementos que están en el llamante pero *
   * no en el parámetro)                                     */
  public void difference (SetIF<E> s);

  /* Devuelve true sii el conjunto parámetro es subconjunto *
   * del llamante                                            */
  public boolean isSubset (SetIF<E> s);
}
```

CONTENEDORES: MULTICONJUNTOS

Equipo Docente de la Asignatura. Departamento LSI.

UNED
ETS de Ingeniería Informática

Tipos Abstractos de Datos estudiados en esta asignatura

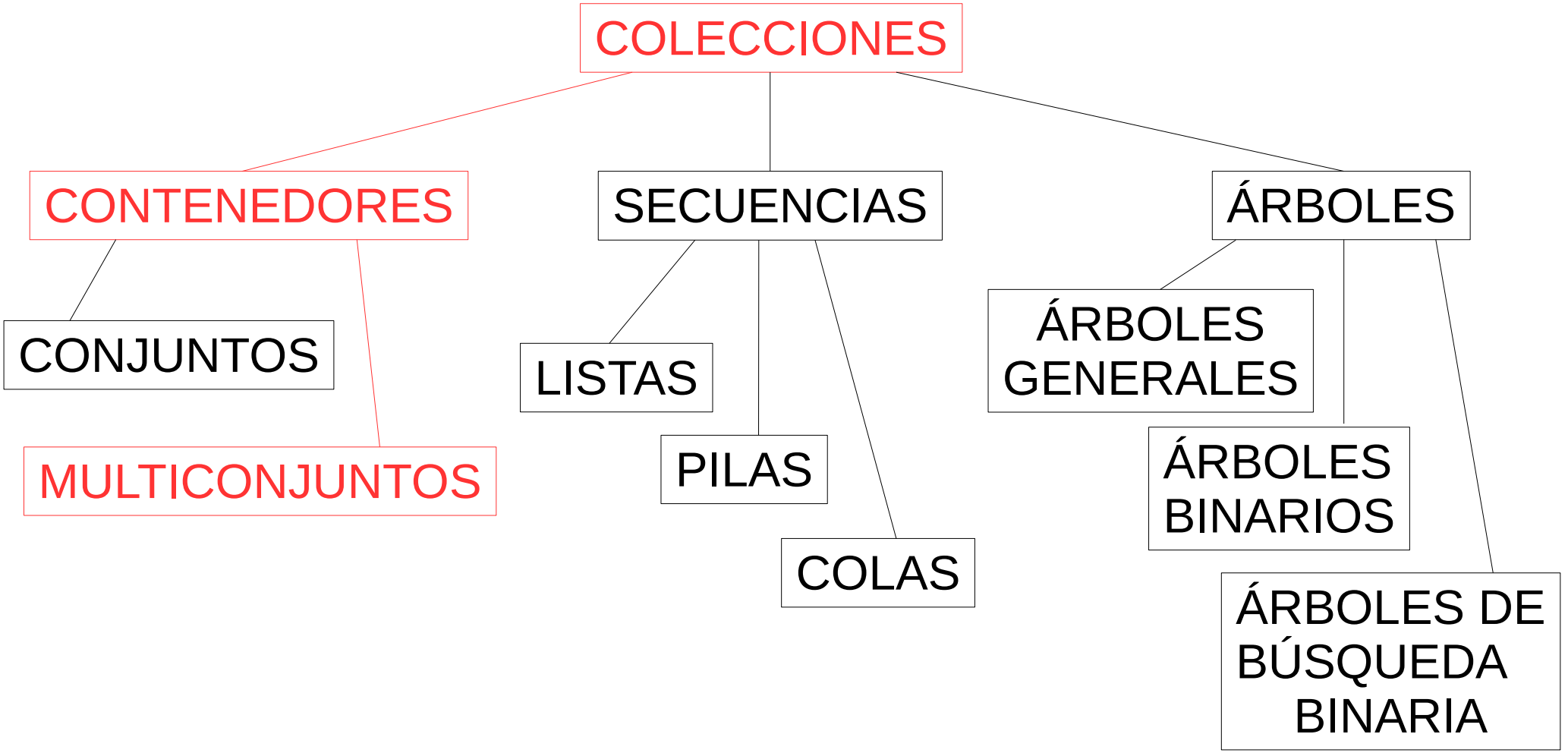


Equipo Docente de la Asignatura. Departamento LSI.

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 MULTICONJUNTOS

- Concepto matemático de multiconjunto finito
  - Puede haber varias instancias de cada elemento
- ¿Qué operaciones hacen falta?
  - Operaciones multiconjuntistas: unión, intersección, diferencia
  - Comprobar si es submulticonjunto de otro
  - Obtener multiplicidad de un elemento
  - Añadir varias instancias de un elemento
  - Eliminar varias instancias de un elemento

Equipo Docente de la Asignatura. Departamento LSI.

UNED
ETS de Ingeniería Informática

ESTRUCTURAS DE DATOS
CONJUNTOS vs MULTICONJUNTOS

Equipo Docente de la Asignatura. Departamento LSI.

UNED
ETS de Ingeniería Informática

ESTRUCTURAS DE DATOS
CONJUNTOS vs MULTICONJUNTOS

1

CONJUNTO

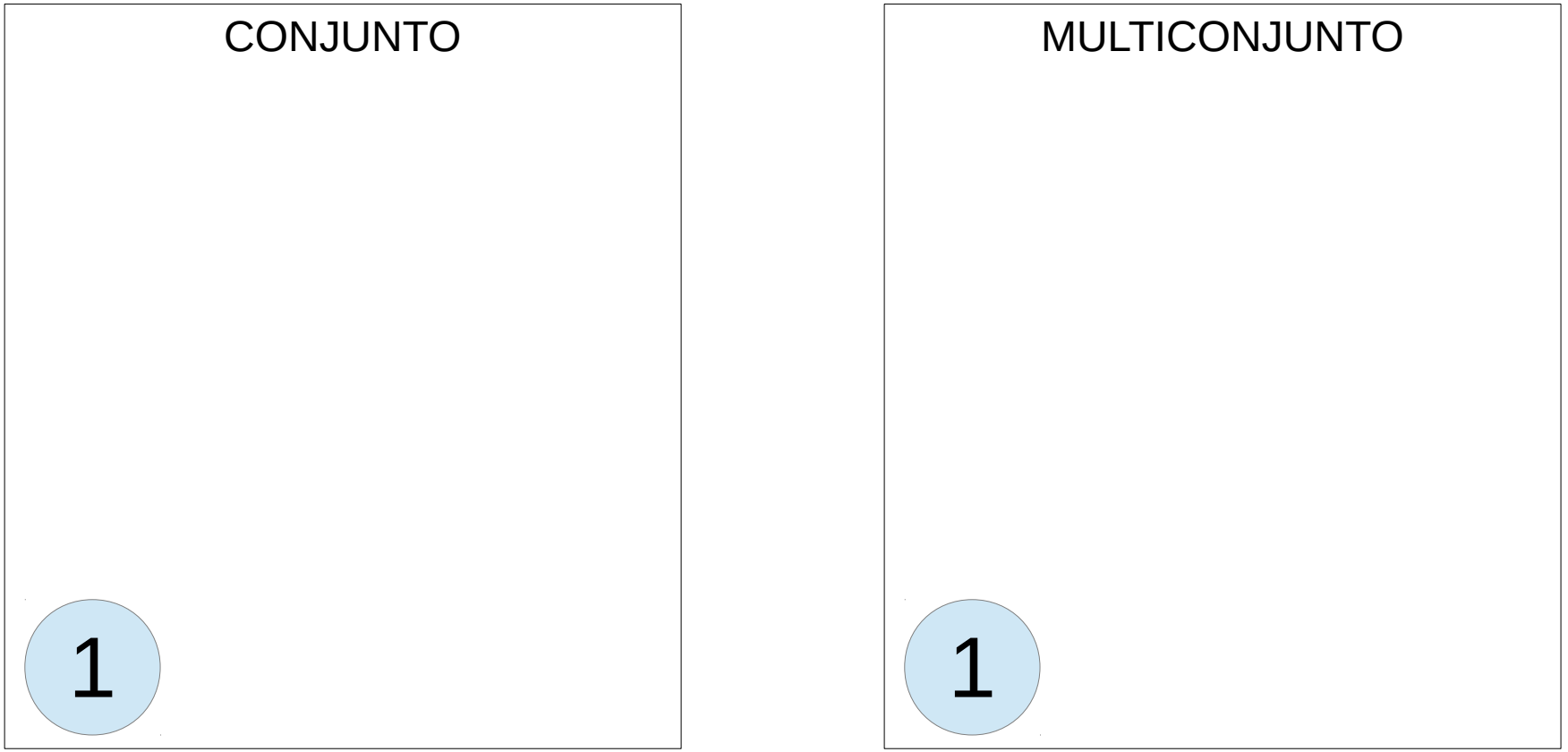
MULTICONJUNTO

Equipo Docente de la Asignatura. Departamento LSI.

UNED

ETS de Ingeniería Informática

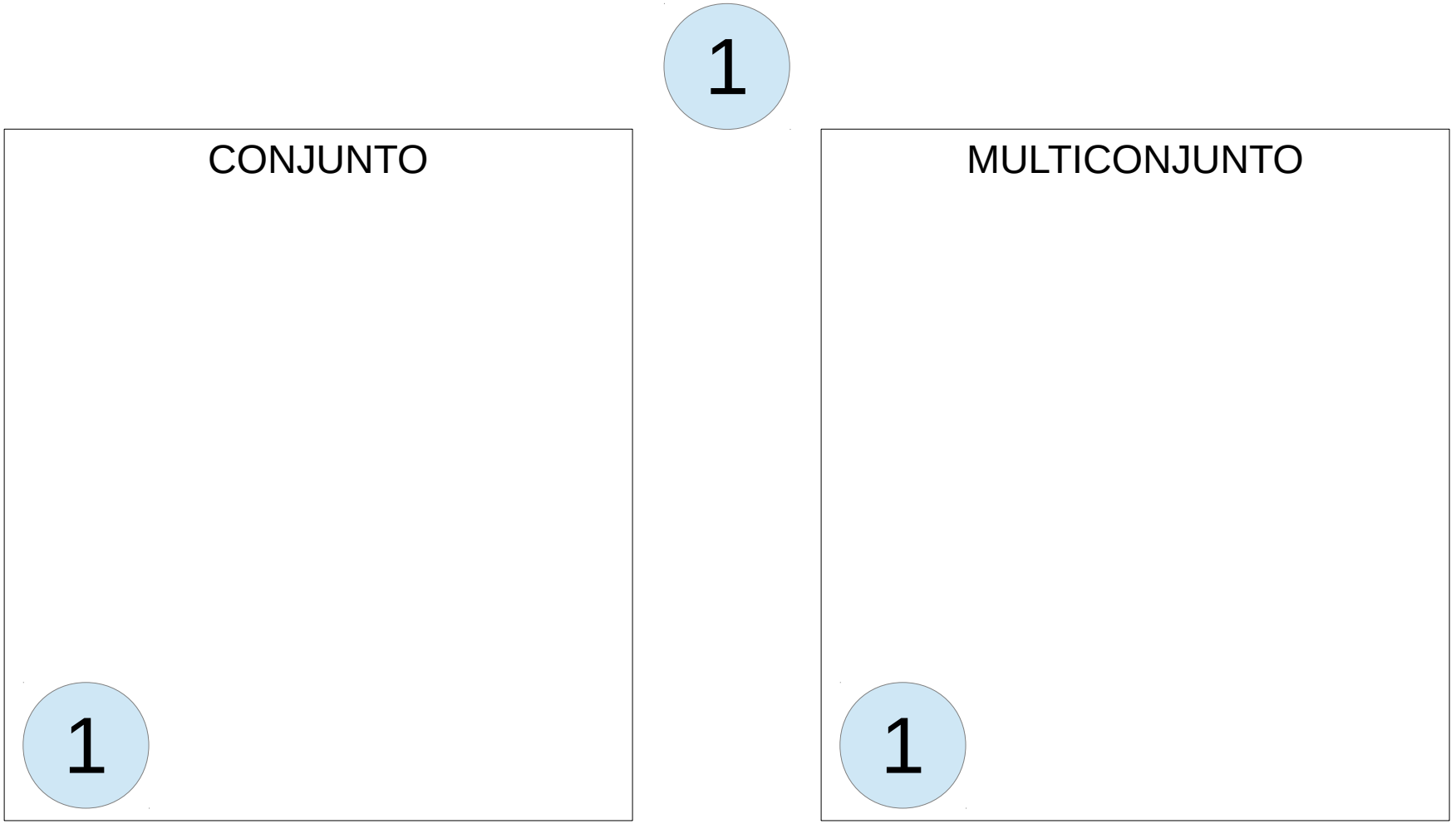
ESTRUCTURAS DE DATOS
CONJUNTOS vs MULTICONJUNTOS



Equipo Docente de la Asignatura. Departamento LSI.

UNED
ETS de Ingeniería Informática

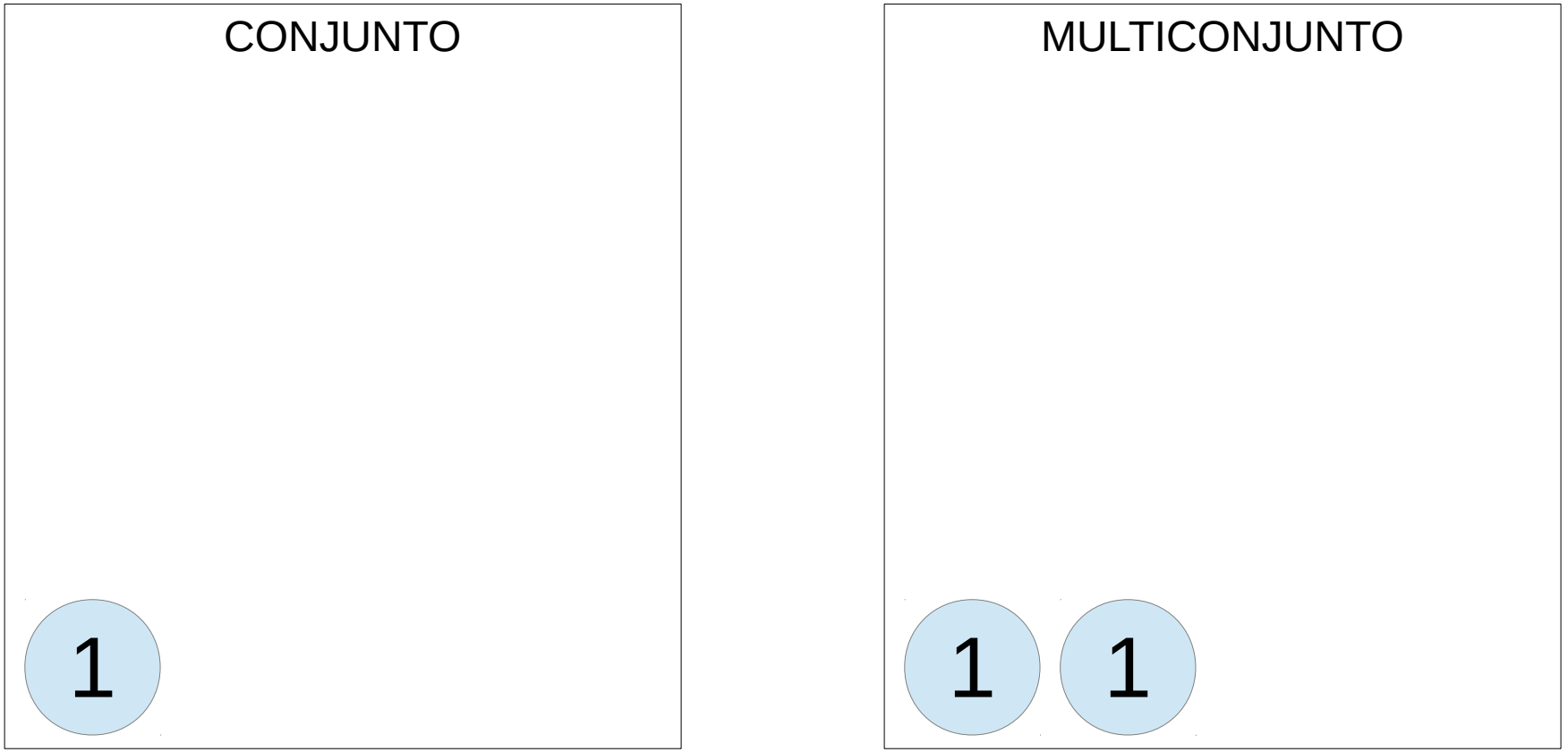
ESTRUCTURAS DE DATOS
CONJUNTOS vs MULTICONJUNTOS



Equipo Docente de la Asignatura. Departamento LSI.

UNED
ETS de Ingeniería Informática

ESTRUCTURAS DE DATOS
CONJUNTOS vs MULTICONJUNTOS



Equipo Docente de la Asignatura. Departamento LSI.

UNED
ETS de Ingeniería Informática

ESTRUCTURAS DE DATOS
MULTICONJUNTOS

```
/* Representa un multiconjunto, que es un contenedor que     *
 * permite almacenar elementos de los que puede haber        *
 * múltiples instancias dentro del multiconjunto.            */
public interface MultiSetIF<E> extends ContainerIF<E> {

  /* Añade varias instancias de un elemento al multiconjunto*
   * @pre:  n > 0 && premult = multiplicity(e)                *
   * @post: multiplicity(e) = premult + n                     */
  public void addMultiple (E e, int n);

  /* Elimina varias instancias de un elemento del             *
   * multiconjunto                                            *
   * @pre: 0<n<=multiplicity(e) && premult=multiplicity(e)    *
   * @post: multiplicity(e) = premult - n                     */
  public void removeMultiple (E e, int n);

                                  ...
```

Equipo Docente de la Asignatura. Departamento LSI.

UNED
ETS de Ingeniería Informática

ESTRUCTURAS DE DATOS
MULTICONJUNTOS

```
/* Devuelve la multiplicidad de un elemento dentro del        *
 * multiconjunto.                                             *
 * @return: multiplicidad de e (0 si no está contenido)      */
public int multiplicity (E e);

/* Realiza la unión del multiconjunto llamante con el         *
 * parámetro                                                  */
public void union (MultiSetIF<E> s);

/* Realiza la intersección del multiconjunto llamante con     *
 * el parámetro                                               */
public void intersection (MultiSetIF<E> s);
```

...

Equipo Docente de la Asignatura. Departamento LSI.

UNED ETS de Ingeniería Informática

ESTRUCTURAS DE DATOS MULTICONJUNTOS

```
/* Realiza la diferencia del multiconjunto llamante con el  *
 * parámetro (los elementos que están en el llamante pero   *
 * no en el parámetro                                       */
public void difference (MultiSetIF<E> s);

/* Devuelve cierto sii el parámetro es un submulticonjunto  *
 * del llamante                                             */
public boolean isSubMultiSet (MultiSetIF<E> s);
}
```

ESTRATEGIAS DE PROGRAMACIÓN Y ESTRUCTURAS DE DATOS

Estructuras de Datos Básicas (I)

Interfaces, Iteradores, Colecciones y Contenedores

Equipo Docente de la Asignatura. Departamento LSI.

UNED ETS de Ingeniería Informática