Algorítmica II
Ingeniería Informática en Sistemas de Información
EPD-8: CONJUNTOS

Introducción

En esta EPD se explorarán los conceptos fundamentales de conjuntos, presentando el Tipo Abstracto de Datos (TAD) de esta colección, que deberá ser usada para resolver los experimentos y problemas propuestos en esta práctica. Una vez conocido el TAD, se expondrá una implementación clásica de los conjuntos basada en vectores.

Conjuntos

Un conjunto (*set*) pude definirse como una colección de elementos en la que no existen duplicados y no se establece ninguna relación posicional concreta entre ellos. Conceptualmente, esta situación sería similar a disponer de una bolsa o caja en la que se introdujeran los elementos, donde no existe ningún tipo de organización ya que no tienen ninguna relación inherente que los una, siendo indiferente el orden en el que los elementos se hayan añadido al conjunto. En la figura 1 podemos ver un ejemplo de cuatro conjuntos de números enteros.

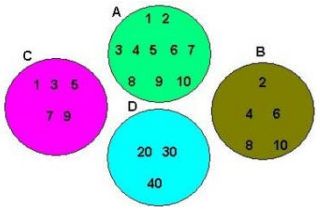

Figura 1. Ejemplos de conjuntos.

Para trabajar con este tipo de colecciones se usará el TAD Conjunto *(SetADT)* que tendrá disponible el conjunto de operaciones representadas en la tabla 1, que servirán para resolver cualquier problema donde pueda ser útil el uso de un conjunto para implementar la solución.

Tabla 1. Operaciones comunes de los Conjuntos.

Operación	Descripción
`add`	Añade un elemento al conjunto
`addAll`	Añade los elementos de un conjunto a otro
`removeRandom`	Elimina un elemento aleatorio del conjunto
`remove`	Elimina un elemento concreto del conjunto
`union`	Combina los elementos de dos conjuntos para crear un tercero
`contains`	Determina si un elemento concreto se encuentra dentro del conjunto
`equals`	Determina si dos conjuntos contienen los mismos elementos
`isEmpty`	Determina si el conjunto está vacío
`size`	Determina el número de elementos del conjunto
`iterator`	Proporciona un iterador para el conjunto
`toString`	Devuelve una representación del conjunto en forma de cadena de caracteres

Los conjuntos pueden ser implementados usando múltiples estructuras de datos. En concreto para esta práctica usaremos la implementación clásica basada en vectores representada en la figura 2 *(ArraySet)*.

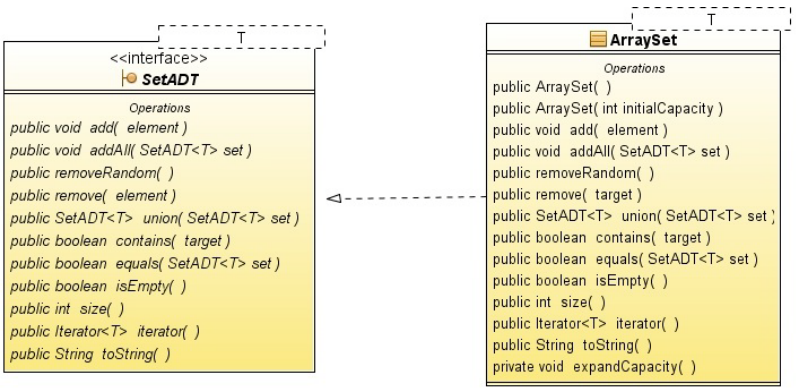

Figura 2. TAD Conjunto con su implementación.

Preparación de la práctica

Todo el material necesario para resolver los experimentos o problemas se encuentra disponible en WebCT. En la carpeta de esta EPD podrá encontrar:

- Enunciado de esta práctica en PDF.
- Fichero Sets.jar
- Fichero con los códigos fuentes de árboles binarios: Sets.zip

Cuando cree un proyecto para resolver un determinado experimento o problema, deberá importar la librería Sets.jar si necesita trabajar con conjuntos.

No olvide añadir donde corresponda la cláusula de importación para hacer uso de los TAD y sus implementaciones:

```
import sets.*;
```

Experimentos

E1. (10 minutos) Cree un proyecto que haciendo uso del TAD Conjunto y de su implementación mediante un vector, introduzca en un primer conjunto todos los caracteres del alfabeto en minúsculas y en un segundo conjunto todos los caracteres dígitos. Cree un nuevo conjunto resultante de realizar la operación de unión entre los dos conjuntos anteriores. Para finalizar mostrará por pantalla los tres conjuntos con los que se ha trabajado en este experimento.

E2. (15 minutos) Cree un proyecto que haciendo uso del TAD Conjunto y de su implementación mediante un vector, cree un primer conjunto con los números enteros menores de 50 que son múltiplos de 2 y un segundo conjunto con los múltiplos de 3 menores de 50. A partir de estos dos conjuntos cree otro conjunto donde se almacenen todos los números naturales menores de 50 que sean múltiplos de 2 o múltiplos de 3. Muestre por pantalla el contenido de los tres conjuntos junto con el número de elementos que los componen, ¿existe alguna relación especial entre el número de elementos de los dos primeros conjuntos y el tercero?, ¿qué propiedad fundamental del TAD Conjunto explica dicho resultado?

Ejercicios

EJ1. (45 minutos) Dados dos conjuntos cualesquiera el método *elementosDiferentes* crea un nuevo conjunto con los elementos que solo aparecen en uno de los dos conjuntos iniciales. Usando conjuntos de números enteros, cree un proyecto que implemente dos soluciones para este método, una iterativa y otra recursiva. Deberá contrastar que partiendo de los mismos conjuntos iniciales la salida de ambos métodos es la misma (para ello use el método *equals* del TAD Conjunto).

EJ2. (35 minutos) Desarrolle un programa que simule el sorteo y comprobación de boletos premiados de la Lotería Primitiva. Inicialmente se procederá a rellenar el bombo del sorteo con las 49 bolas necesarias y se extraerán al azar 6 bolas diferentes que conformarán la combinación ganadora. A continuación el programa solicitará al usuario los 6 números marcados en su boleto, comprobará los aciertos obtenidos y mostrará por pantalla un resumen con la combinación ganadora, el boleto del usuario con los aciertos obtenidos.

Problemas

P1. (30 minutos). La operación *diferencia* toma como parámetros dos conjuntos, A y B, y devuelve un tercer conjunto C que es el resultado de eliminar del conjunto A los elementos que están en B. Cree un proyecto donde se desarrolle un método que implemente dicha operación.

P2. (30 minutos) La operación *intersección* toma como parámetros dos conjuntos, A y B, y devuelve un tercer conjunto C con los elementos que tienen en común ambos conjuntos. Cree un proyecto donde se desarrolle un método que implemente dicha operación.

P3. (60 minutos) En WebCT tiene disponible el fichero *Sets.zip* con el código fuente del TAD Conjunto con su implementación *ArraySet*. Modifique dicha clase para que el usuario pueda controlar la capacidad del conjunto. Elimine la característica de expansión automática del vector. La clase modificada debe generar una excepción *FullSetException* cuando se añada un elemento a un conjunto que ya esté lleno. Añada un método denominado *isFull* que devuelva un valor booleano que diferencie si el conjunto está lleno o no. Cree también un método que el usuario pueda invocar para expandir la capacidad del conjunto, añadiendo un número concreto de elementos más.

P4. (120 minutos) Cree un proyecto con una nueva implementación del TAD Conjunto *(SetADT)* basada en el TAD Árbol Binario Ordenado *(BinarySearchTreeADT)*.