UNIVERSIDADE DA CORUÑA

FACULTADE DE INFORMÁTICA
Departamento de Computación
Programación II

Especificación de Tipos Abstractos de Datos: Ejercicio resuelto

1. PROBLEMA: Se necesita un tipo de dato para representar y manejar colecciones de elementos no ordenados sin considerar duplicados. Definir un TAD Conjunto que incluya las operaciones más típicas.

SOLUCIÓN

El primer paso consiste en definir el conjunto de valores del nuevo tipo de dato.

Valores: Colección de elementos no ordenados sin duplicados. Si el número de elementos es cero, el conjunto se dice vacío.

El segundo paso consiste en nombrar las distintas operaciones necesarias para manipular el nuevo tipo de dato. Normalmente dependerá del problema y será lo más completa posible. En nuestro caso, consideraremos sólo las siguientes operaciones:

a. Crear un conjunto vacío
b. Determinar si un conjunto es o no vacío
c. Añadir o eliminar elementos de un conjunto
d. Determinar si un elemento pertenece o no a un conjunto
e. Calcular el número de elementos de un conjunto
f. Realizar la unión, intersección y diferencia de conjuntos
g. Determinar si un conjunto está incluido en otro

El tercer paso consiste en clasificar este conjunto de operaciones relacionadas con el nuevo tipo de dato en tres categorías: *constructoras y destructoras*—aquellas cuyo resultado es de tipo conjunto—y *observadoras*—aquellas que tienen uno o más argumentos de tipo conjunto y cuyo resultado no es de tipo conjunto. A su vez, dividiremos las operaciones constructoras en *generadoras*—necesarias para generar todos los valores del tipo, tal que excluyendo alguna de ellas ya no es posible generar todos esos valores—y *modificadoras*—el resto. Para cada una de estas operaciones detallaremos su *objetivo*, el tipo de dato de los parámetros de entrada y/o salida y las *precondiciones* y *poscondiciones* si las hubiera.

Operaciones: Las operaciones anteriormente citadas se organizan en tres categorías generales: constructoras (generadoras y modificadoras), destructoras y observadoras.

- Generadoras. Todo conjunto finito puede construirse a partir del vacío, añadiendo sucesivamente elementos de uno en uno[1].

  ConjuntoVacio → Conjunto
  {*Objetivo*: Crea un conjunto vacío
  Salidas: Un conjunto inicializado}

  Añadir (Elemento, Conjunto) → Conjunto
  {*Objetivo*: Añade un elemento al conjunto
  Entradas: Un conjunto y un elemento
  Salidas: El conjunto con el nuevo elemento si no existía
  Precondición: Conjunto inicializado}

- Modificadoras. Conjunto de operaciones constructoras no generadoras.

  Union (Conjunto1, Conjunto2) → Conjunto3
  {*Objetivo*: Realiza la operación de Unión de conjuntos
  Entradas: Dos conjuntos
  Salidas: Un nuevo conjunto con los elementos comunes y no comunes de ambos conjuntos
  Precondición: Conjuntos inicializados}

  Interseccion (Conjunto1, Conjunto2) → Conjunto3
  {*Objetivo*: Realiza la operación de Intersección de conjuntos
  Entradas: Dos conjuntos
  Salidas: Un nuevo conjunto con los elementos comunes de ambos conjuntos
  Precondición: Conjuntos inicializados}

  Diferencia (Conjunto1, Conjunto2) → Conjunto3
  {*Objetivo*: Realiza la operación de Diferencia de conjuntos
  Entradas: Dos conjuntos
  Salidas: Un nuevo conjunto con los elementos del primer conjunto que no pertenecen al segundo conjunto
  Precondición: Conjuntos inicializados}

[1] De igual manera se puede crear a partir del conjunto vacío, el conjunto unitario y la unión de conjuntos.

- Destructoras. Es aquel grupo de operaciones que permiten eliminar ejemplares del tipo de dato. Devuelven valores del mismo tipo.

  Eliminar (Elemento, Conjunto) → Conjunto
  {*Objetivo*: Elimina un elemento del conjunto
  Entradas: Un conjunto y un elemento
  Salidas: El mismo conjunto sin el elemento si éste existía
  Precondición: Conjunto inicializado}

- Observadoras. Es aquel grupo de operaciones con al menos un argumento del tipo especificado pero que devuelven valores de otro tipo.

  EsConjuntoVacio (Conjunto) → Booleano
  {*Objetivo*: Determina si el conjunto contiene o no elementos
  Entradas: Un conjunto
  Salidas: Verdadero si el conjunto está vacío, falso en caso contrario
  Precondición: Conjunto inicializado}

  Pertenece (Elemento, Conjunto) → Booleano
  {*Objetivo*: Determina si un elemento está incluido en el conjunto
  Entradas: Un conjunto y un elemento
  Salidas: Verdadero si existe el elemento en el conjunto, falso en caso contrario
  Precondición: Conjunto inicializado}

  Inclusion (Conjunto1, Conjunto2) → Booleano
  {*Objetivo*: Determina si un conjunto esta incluido en otro
  Entradas: Dos conjuntos
  Salidas: Verdadero si el primer conjunto está incluido en el segundo, falso en caso contrario
  Precondición: Conjuntos inicializados}

  Cardinal (Conjunto) → Natural
  {*Objetivo*: Calcula el número de elementos del conjunto
  Entradas: Un conjunto
  Salidas: Número de elementos que contiene el conjunto
  Precondición: Conjunto inicializado}