Universidad Rey Juan Carlos
1º GII/GII+ADE (Vicálvaro)

Asignatura: Estructuras de Datos

HOJA DE PROBLEMAS Nº 2:
Tipos Abstractos de Datos y Especificaciones Algebraicas

1) Especificar una extensión del TAD *Naturales* que incluya las operaciones siguientes:

   - *EsImpar*, operación booleana que decide si un número natural es impar.
   - *EsMenor*, operación booleana que decide si un número natural es menor (estrictamente) que otro.
   - *Resta*, operación que permite realizar la resta de dos números naturales. Téngase en cuenta que la operación sólo estará definida si el segundo es menor o igual que el primero.
   - *Mcd*, operación que recibe dos números naturales no nulos y devuelve su máximo común divisor. Se recuerda a continuación un algoritmo recursivo para calcular el máximo común divisor de dos números positivos (se supondrá disponible el operador MOD que calcula el resto de una división entera).

$$Mcd(a,b) = \begin{cases} b & \text{si } a \text{ MOD } b = 0 \\ Mcd(b, a \text{ MOD } b) & \text{e.o.c.} \end{cases}$$

2) Especificar una extensión del TAD *Booleanos1* que incluya las operaciones *Xor* (disyunción exclusiva) e *Implicación* (se recuerda que la operación lógica '*a* implica *b*' se define cómo 'No *a* o *b*').

3) Especificar una extensión del TAD *Complejos* que incluya las operaciones siguientes:

   - *EsImaginarioPuro*, operación booleana que decide si un número complejo es imaginario puro (e.d., no tiene parte real).
   - *EsReal*, operación booleana que decide si un número complejo es un número real.
   - *EsNulo*, operación booleana que decide si un número complejo es nulo.
   - *Argumento*, operación que recibe como entrada un número complejo no nulo y devuelve su argumento. Se recuerda a continuación el valor del argumento de un número complejo *a+bi* no nulo (número real comprendido entre –Pi y Pi):

arctg(b/a)	*si a>0*
arctg(b/a) + pi	*si a<0 y b>=0*
arctg(b/a) - pi	*si a<0 y b<0*
pi/2	*si a=0 y b>0*
-pi/2	*si a=0 y b<0*

4) Especificar una extensión del TAD *Círculos* que incluya las operaciones siguientes:

   - *Área*, operación que devuelve el área del círculo.
   - *EstáEnPerímetro*, operación booleana que recibe como entradas un punto y un círculo y devuelve cierto si y sólo si el punto pertenece al perímetro del círculo.
   - *Tangentes*, operación booleana que recibe como entrada dos círculos y devuelve cierto si y sólo si los círculos son tangentes. Se recuerda que dos círculos pueden ser tangentes *interiormente* (cuando la distancia entre sus centros es igual al valor absoluto de la diferencia de sus radios) o *exteriormente* (cuando la citada distancia es igual a la suma de los radios).

- *Secantes*, operación booleana que recibe como entrada dos círculos y devuelve cierto si y sólo si los círculos son secantes, es decir, la distancia entre sus centros está comprendida, estrictamente, entre el valor absoluto de la diferencia de los radios y la suma de éstos.
- *Disjuntos*, operación booleana que recibe como entrada dos círculos y devuelve cierto si y sólo si los círculos son disjuntos, en el sentido de que o bien son exteriores (la distancia entre sus centros es mayor que la suma de los radios) o bien uno está dentro del otro (la distancia entre sus centros es menor que el valor absoluto de la diferencia de sus radios).

5) Especificar una extensión del TAD *Vectores* que incluya las operaciones siguientes:

- *EsVectorVacio*, operación booleana que recibe un vector y devuelve cierto si y sólo si el vector no tiene ninguna posición asignada.
- *MáximaPos*, operación genérica tal que, dado un vector no vacío, devuelve la mayor de las posiciones que están asignadas. El parámetro formal de dicha operación será una operación booleana *Menor* que permite comparar dos elementos del tipo *TipoRango*.
- *Permuta*, operación que recibe como entradas un vector y dos elementos de *TipoRango* y, si estos dos elementos se corresponden con posiciones asignadas en el vector, devuelve un vector con los valores de ambas posiciones intercambiados.

6) Especificar algebraicamente un TAD *Fracciones* para trabajar con números fraccionarios. Dicho TAD consta de la operación constructora generadora siguiente:

$$CrearFracción: Entero \times Entero\text{-}\{0\} \rightarrow TipoFraccion$$

Téngase en cuenta que el conjunto de generadoras (en este caso compuesto por una única operación) no es libre, puesto que permite construir términos distintos que sin embargo denotan la misma fracción: por ejemplo CrearFracción(4,-2), CrearFracción(-8,4), CrearFracción(-2,1) y CrearFracción(2,-1) son términos distintos que denotan el mismo valor. Será necesario por lo tanto incluir el apartado "Ecuaciones entre generadoras", donde se reflejará esta equivalencia entre términos distintos. La forma más sencilla de hacerlo es especificando que toda fracción es igual a la "fracción canónica" representativa de su clase, que podría interpretarse como sigue:

- en el caso de las fracciones con 0 en el numerador, la fracción canónica sería la fracción con 0 en el numerador y 1 en el denominador;
- en el resto de casos, la fracción canónica sería la única fracción de la clase que es irreducible y tiene el signo en el numerador (en el ejemplo anterior, el término canónico sería CrearFracción(-2,1)).

La especificación deberá incluir las siguientes operaciones:

- *NumeradorCanónico*, operación que devuelve el numerador del término canónico asociado.
- *DenominadorCanónico*, operación que devuelve el denominador del término canónico asociado.
- *EsEntero*, operación booleana que decide si una fracción es un número entero.
- *EsFraccionNula*, operación booleana que decide si una fracción es nula.
- *Menor*, operación booleana que recibe dos fracciones y decide si la primera es menor que la segunda.
- *Suma*, *Resta*, *Multiplicación* y *División*, operaciones que permiten realizar operaciones aritméticas entre fracciones. Téngase en cuenta la posible parcialidad de alguna de estas operaciones.

Nota: en la especificación anterior se supondrán disponibles las siguientes operaciones para el tipo de datos *Entero*: *abs* (valor absoluto), *Mcd* (máximo común divisor para enteros positivos), y *Signo* (que devuelve –1 si el entero pasado como parámetro es negativo y 1 en caso contrario).

7) Se desea especificar algebraicamente el TAD *TipoHoraDigital*, el cual representa la hora mediante tres valores, pertenecientes, respectivamente, a los tipos de datos *TipoHora* (subrango de los números naturales comprendidos entre 0 y 23), *TipoMinuto* y *TipoSegundo* (subrangos de los números naturales comprendidos entre 0 y 59).

Para ello se tendrá en cuenta que cualquier hora digital puede construirse mediante la siguiente operación:

ConstruirHora: TipoHora $\times$ *TipoMinuto* $\times$ *TipoSegundo* $\rightarrow$ *TipoHoraDigital*

Se pide realizar la especificación algebraica completa del TAD *TipoHoraDigital*, incluyendo, además de *ConstruirHora*, las siguientes operaciones, cuyo significado se indica entre comentarios:

Segundos: TipoHoraDigital $\rightarrow$ *TipoSegundo*
(* devuelve los segundos *)
Suma: TipoHoraDigital x TipoHoraDigital $\rightarrow$ *TipoHoraDigital*
(* dadas dos horas digitales *hora1* y *hora2*, devuelve el resultado de sumar a *hora1* las horas, minutos y segundos de *hora2* *)
EnPunto: TipoHoraDigital $\rightarrow$ *Booleano*
(* decide si una hora digital es justamente en punto *)
Horas: TipoHoraDigital $\rightarrow$ *TipoHora*
(* devuelve las horas*)
SegundosAHora: Natural $\rightarrow$ *TipoHoraDigital*
(* dada una cantidad de segundos, devuelve la hora digital correspondiente *)
Incrementar: TipoHoraDigital $\times$ *Natural* $\rightarrow$ *TipoHoraDigital*
(* dada una hora digital y un número de segundos devuelve la hora digital incrementada con esa cantidad de segundos *)
Minutos: TipoHoraDigital $\rightarrow$ *TipoMinuto*
(* devuelve los minutos *)
Distancia: TipoHoraDigital $\times$ *TipoHoraDigital* $\rightarrow$ *Entero*
(*dadas dos horas digitales *hora1* y *hora2*, devuelve el número total de segundos entre ambas; en caso de que *hora1* sea posterior a *hora2* el resultado será negativo*)
EsPosterior: TipoHoraDigital $\times$ *TipoHoraDigital* $\rightarrow$ *Booleano*
(* dadas dos horas digitales *hora1* y *hora2* devuelve cierto si *hora1* es posterior a *hora2*; en otro caso, devuelve falso *)

Nota: téngase en cuenta que los tipos de datos *TipoHora*, *TipoMinuto* y *TipoSegundo*, al ser todos ellos subrangos de los números naturales, disponen de todas las operaciones de éstos (entre otras las operaciones aritméticas estándar, operadores de comparación, y los operadores *div* y *mod* para obtener, respectivamente, el cociente y el resto de una división entera).