Estructuras de Datos y Algoritmos

Grados en Ingeniería Informática, de Computadores y del Software

Examen Parcial, 11 de Febrero de 2013.

1. Diseño iterativo (4 puntos) Especificar, diseñar, verificar y calcular el coste de un algoritmo que, dado un vector v de enteros que puede ser vacío ($n \geq 0$) y que tan solo puede contener los valores 0 y 1, devuelva un booleano que indique si el vector tiene la forma:

$$\boxed{1 \mid 1 \mid \cdots \mid 1 \mid 1 \mid 0 \mid 0 \mid \cdots \mid 0 \mid 0}$$

Las dos zonas de ceros y unos pueden tener cualquier longitud, incluida la nula.

Sugerencia (no penaliza si no se sigue): la verificación puede resultar más fácil si se elige un invariante de la forma:

$$\boxed{1 \mid \cdots \mid 1 \mid \ ? \ ? \ ? \ ? \ ? \ \mid 0 \mid \cdots \mid 0}$$

2. Diseño recursivo (4 puntos) Se define el *histograma* de un vector v de enteros como otro vector w que contiene en la posición i la suma $v[0]+\ldots+v[i]$. Así, por ejemplo, el vector $\{1, 2, 4, 3, -2\}$ tendría como histograma $\{1, 3, 7, 10, 8\}$.

Se pide especificar, diseñar, demostrar la corrección y calcular el coste de un algoritmo recursivo, lo más eficiente posible, que dado un vector de enteros que puede ser vacío devuelva su histograma.

3. Diseño TADs (2 puntos) Diseñar un TAD *Polinomio* que permita manejar polinomios con coeficientes enteros en una indeterminada. Ejemplos de polinomios serían $3x^4+1$, $-2x^{99}+5x^2$, ó $x^{507}+x^{200}+x$. El tipo *Polinomio* deberá permitir, además de la construcción de polinomios, su suma, resta y multiplicación (que devolverán nuevos polinomios); su evaluación para una x concreta; y la posibilidad de verificar si un polinomio está vacío o es igual a otro dado.

Se pide:

- Elegir un *tipo representante* para el TAD eficiente en tiempo y espacio, suponiendo que los polinomios están formados por k o menos monomios (elementos de la forma $c \cdot x^n$), donde tanto c como n pueden llegar a ser muy grandes (aunque quepan en un `int` estándar). La constante k debe formar parte de la implementación de tu TAD.
- Dar el *invariante de la representación* y la *relación de equivalencia* de la representación elegida.
- Implementar el archivo `polinomio.h` con la definición de la clase. Se debe incluir las operaciones públicas necesarias para manejar el tipo, ya sean constructoras, observadoras o modificadoras; y la parte privada con la representación. Para cada operación debe darse su precondición y postcondición.

Nota: No hace falta que implementes ninguna de estas operaciones; basta con que las declares y describas correctamente.