Estructuras de Datos y Algoritmos

Grados en Ingeniería Informática, de Computadores y del Software (grupo A)

Examen Final, 9 de Septiembre de 2014.

1. (3 puntos) Consideremos un vector V[N] con 0 < N de números enteros, cuyos valores se han obtenido aplicando una rotación sobre un vector ordenado en orden estrictamente decreciente. Implementa un algoritmo que calcule el mínimo del vector con una complejidad $\mathcal{O}(\log n)$. El número de elementos sobre los que se aplica la rotación para obtener el vector de entrada cumple 0 < elementos rotados < N y no se conoce.

Ejemplo: un posible vector de entrada sería el vector 70 55 13 4 100 80 obtenido desplazando los dos primeros elementos del vector 100 80 70 55 13 4 al final del mismo.

2. (3 puntos) Dada la siguiente serie

$$x + \frac{x^3}{3} + \frac{x^5}{5} + \frac{x^7}{7} + \dots$$

donde x es un número real tal que |x| < 1, y dado un entero $n \ge 0$, especifica, diseña y verifica o especifica y deriva, un algoritmo iterativo de coste lineal para calcular la suma de los n primeros términos de la serie. No se puede considerar que la operación de cálculo de una potencia se realiza en tiempo constante. Justificar el coste de la función implementada.

- **3.** (4 puntos) Estás trabajando en un nuevo videojuego llamado *CiudadMatic*: un simulador de ciudades, llenas de edificios de distinto tipo. Será posible construir y reparar estos edificios gastando dinero, y al final de cada turno (después de haber construido y reparado tantos edificios como se quiera), recaudar los impuestos que generen. Necesitarás implementar las siguientes operaciones:
 - CiudadMatic: inicializa una nueva ciudad vacía, con el dinero que se pase como argumento disponible para ser gastado.
 - nuevo Tipo: añade un nuevo tipo de edificio al sistema, con un identificador proporcionado por el usuario (por ejemplo, "bar"), un coste de construcción, una cantidad de impuestos generada por turno, y una calidad de base (máximo de turnos sin reparar). No devuelve nada.
 - inserta Edificio: dado el nombre de un edificio (por ejemplo, "El Bar de Moe"), y el identificador de su tipo, y asumiendo que se disponga del dinero necesario para construirlo, añade el edificio a la ciudad y resta su coste de construcción del dinero disponible. No devuelve nada.
 - repara Edificio: repara el edificio cuyo identificador se pase como argumento a "como recién construido", a un coste del 10 % del coste de construcción (descartando los decimales), independientemente de lo estropeado que estuviese. No devuelve nada.
 - finTurno: todos los edificios construidos generan los impuestos que les corresponden por su tipo. Después, todos se estropean por un punto de calidad, y aquellos que lleguen a 0 son derribados y eliminados de la ciudad. Devuelve el dinero total disponible para el nuevo turno (impuestos generados + dinero no gastado del turno anterior).

• lista Edificios: dado un identificador de tipo de edificio, devuelve una lista con los edificios de ese tipo que están actuamente construidos, por orden de antigüedad (primero el más viejo).

Desarrolla en C++ una implementación de la clase *CiudadMatic* basada en otros TADs conocidos, optimizando la complejidad temporal de las operaciones. En cada operación, indica también, de forma razonada, su complejidad.