Estructuras de Datos y Algoritmos

Grados en Ingeniería Informática

Examen Final Septiembre, 12 de septiembre de 2017

Nombre: _____ **Grupo:** _____

Laboratorio: _____ **Puesto:** _____ **Usuario de DOMjudge:** _____

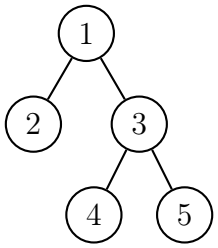
4. (2 puntos) Se desea contar el número de elementos de un vector que cumplen que la suma de los elementos a su izquierda es menor estricto que la suma de los elementos a su derecha. Se pide especificar una función que resuelva el problema; implementar como cuerpo de dicha función un algoritmo iterativo eficiente que resuelva el problema; escribir los invariantes y funciones de cota que permitan demostrar la corrección del algoritmo propuesto, y justificar adecuadamente el coste asintótico en el caso peor del algoritmo.

Para probar la función, el programa principal resolverá varios casos. Este código, que **no** puede ser modificado, se encuentra en el fichero `main4.cpp`, siendo la función `resolver` la que se pide en el enunciado.

5. (3 puntos) Dado un conjunto de $n > 0$ números enteros (n par), se desea dividir el conjunto en dos subconjuntos con $n/2$ elementos cada uno de forma que el valor absoluto de la diferencia entre las sumas de los elementos de los subconjuntos sea mínima. Implementar un algoritmo de vuelta atrás que devuelva una solución óptima y el valor absoluto de su diferencia.

Para probar la función, el programa principal resolverá varios casos e imprimirá por pantalla el valor de las soluciones óptimas. Este código, que **no** puede ser modificado, se encuentra en el fichero `main5.cpp`, siendo la función `resolver` la que se pide en el enunciado.

6. (2 puntos) Implementa una función `hojas` que, dado un árbol binario de enteros y un número entero no negativo k, determine el número de hojas cuya profundidad es mayor que k. Por ejemplo, para el siguiente árbol



la función devolvería 3 si $k = 0$ o $k = 1$, devolvería 2 si $k = 2$ y devolvería 0 si $k \geq 3$.

Aparte de implementar esta función, debes indicar la complejidad de la misma.

El fichero `main6.cpp` contiene ya el código necesario para este ejercicio, salvo la función `hojas`.

7. (3 puntos) La DGT nos ha pedido ayuda para gestionar el *carnet por puntos*. Los conductores están identificados de manera unívoca por su DNI y la cantidad de puntos de un conductor está entre 0 y 15 puntos inclusives. La implementación del sistema se deberá realizar como un TAD `carnet_puntos` con las siguientes operaciones:

- `nuevo(dni)`: Añade un nuevo conductor identificado por su `dni` (un `string`), con 15 puntos. En caso de que el dni esté duplicado, la operación lanza una excepción `ExcepcionTAD` con mensaje `Conductor duplicado`.

- `quitar(dni, puntos)`: Le resta puntos a un conductor tras una infracción. Si a un conductor se le quitan más puntos de los que tiene, se quedará con 0 puntos. En caso de que el conductor no exista, lanza una excepción `ExcepcionTAD` con mensaje `Conductor inexistente`.
- `consultar(dni)`: Devuelve los puntos actuales de un conductor. En caso de que el conductor no exista, lanza una excepción `ExcepcionTAD` con mensaje `Conductor inexistente`.
- `cuantos_con_puntos(puntos)`: Devuelve cuántos conductores tienen un determinado número de puntos. En caso de que el número de puntos no esté entre 0 y 15 lanza una excepción `ExcepcionTAD` con mensaje `Puntos no validos`.

La implementación de las operaciones debe ser lo más eficiente posible. Por tanto, debes elegir una representación adecuada para el TAD, implementar las operaciones y justificar la complejidad resultante.

El programa principal que prueba el nuevo TAD se encuentra en el fichero `main7.cpp`, que **no** puede ser modificado.

Normas de realización del examen

1. Debes programar soluciones para cada uno de los ejercicios, probarlas y entregarlas en el juez automático accesible en la dirección `http://exacrc/domjudge/team`.

2. Escribe comentarios que expliquen tu solución, justifiquen por qué se ha hecho así y ayuden a entenderla. Calcula la complejidad de todas las funciones que implementes.

3. En el juez te identificarás con el nombre de usuario y contraseña que has recibido al comienzo del examen. El nombre de usuario y contraseña que has estado utilizando durante la evaluación continua **no** son válidos.

4. Escribe tu **nombre y apellidos** en un comentario en la primera línea de cada fichero que subas al juez.

5. Descarga el fichero `http://exacrc/DocumA_SEP.zip` que contiene material que debes utilizar para la realización del examen (implementación de las estructuras de datos, ficheros con código fuente y ficheros de texto con algunos casos de prueba de cada ejercicio del enunciado).

6. Tus soluciones serán evaluadas por el profesor independientemente del veredicto del juez automático. Para ello, el profesor tendrá en cuenta **exclusivamente** el último envío que hayas realizado de cada ejercicio.