Algoritmos y Estructuras de Datos (VJ1215) - Universitat Jaume I

Examen final - 2024/2025 - Primera parte
Recuperación del segundo examen parcial

5 de junio de 2025

Nombre:

El examen final consta de dos partes, cada una con una duración de dos horas. La primera parte está destinada a aquellos estudiantes que han solicitado intentar mejorar la calificación obtenida en cualquiera de los exámenes parciales, renunciando a la misma. Este segundo ejercicio de la primera parte está dirigido a quienes buscan mejorar la nota del segundo examen parcial.

La prueba es individual. No puedes consultar libros, apuntes ni dispositivos electrónicos. Escribe tu solución empleando el lenguaje C++, y no olvides los costes que se piden. Al finalizar entrega tu solución junto con el enunciado (no es necesario que entregues todas las hojas). Pon tu nombre en cada hoja que entregues.

Ejercicio 2 — 1,5 puntos

Estamos utilizando un árbol binario de búsqueda (no AVL) para realizar una implementación del Tipo Abstracto de Datos **Conjunto**. Contamos con los siguientes atributos, que tienen el significado habitual. No puedes añadir otros atributos en `Conjunto` ni en `Conjunto::Nodo` (los puntos suspensivos corresponden a posibles declaraciones de métodos):

```
class Conjunto {
   struct Nodo {
      float dato;
      Nodo * izquierdo;
      Nodo * derecho;
      ...
   };
   Nodo * raiz;
   int talla;
   ...
 public:
   ...
};
```

a) **[0,75 puntos] Eliminación de nodos profundos sin bucles**

Añade a la clase `Conjunto` el siguiente método:

```
void eliminarProfundos(int p)
```

Este método debe eliminar todos los nodos cuya profundidad sea mayor o igual que `p`, asumiendo que `p` es mayor o igual que 0.

Ten en cuenta las siguientes consideraciones:

- No confundas profundidad con altura. La profundidad de un nodo es la distancia desde la raíz hasta ese nodo. Si existen, la raíz está a profundidad 0, sus hijos están a profundidad 1, sus nietos están a profundidad 2, y así sucesivamente.
- Si el árbol está vacío o su profundidad es menor que `p`, no debe modificarse.
- No puedes utilizar bucles.
- Puedes emplear otras funciones auxiliares, siempre que las implementes y lo hagas también sin usar ningún bucle.

- Tu solución debe ser eficiente.
- No olvides actualizar la talla del conjunto si cambia.

b) **[0,75 puntos] Eliminación de nodos profundos sin recursión**

Implementa nuevamente el método del apartado anterior, pero esta vez sin utilizar recursión, pudiendo utilizar bucles.

Consideraciones adicionales:

- Puedes utilizar otras funciones auxiliares, siempre que las implementes y lo hagas sin utilizar recursión.
- En este apartado, puedes hacer uso en C++ de `stack`, `queue` y/o `priority_queue`, sin necesidad de implementarlas.
- Si no recuerdas los nombres exactos de sus operaciones básicas, puedes escribirlos en castellano.

Indica cuáles son los siguientes costes de tus dos soluciones en función de n, donde n es la cantidad de nodos del árbol. Justifica solamente a qué se debe ese coste espacial y cuándo se puede dar el peor caso.

	Apartado a	**Apartado b**
Coste temporal en el peor caso	$O($ $)$	$O($ $)$
Coste espacial en el peor caso sin contar el árbol	$O($ $)$	$O($ $)$