# Algoritmos y Estructuras de Datos II
# Recuperatorio del segundo parcial – 23 de Noviembre 2016

**Aclaraciones**

- El parcial es **a libro abierto**.
- Cada ejercicio debe entregarse **en hojas separadas**.
- Incluir en cada hoja el **número de orden asignado, número de hoja, número total de hojas, apellido, nombre y número de Libreta Universitaria**.
- Al entregar el parcial, completar el resto de las columnas en la planilla.
- Cada ejercicio se calificará con **P**romocionado, **A**probado, **R**egular, o **I**nsuficiente.
- El parcial completo está aprobado si el primer ejercicio tiene al menos **A**, y entre los ejercicios 2 y 3 hay al menos una **A**. Para más detalles, ver "Información sobre la cursada" en el sitio Web.

## Ej. 1. Diseño

Se nos encargó el diseño del nuevo Sistema Automático de Facturación Acumulada (SAFA). Dicho sistema recolecta los datos de todas las facturas pagadas mediante el sistema electrónico y genera estadísticas de la cantidad de facturas expedidas y la cantidad de plata electrónica que ingresó al sistema en un determinado período. Al agregar una factura al sistema se especifica su precio y además un natural que representa el momento en que fue expedida esa factura. Dado que el sistema de pago electrónico no se sincroniza al instante, es posible que las facturas se agreguen al sistema no ordenadas cronológicamente. Al consultar, siempre se da un intervalo de tiempo y se debe responder la cantidad de facturas expedidas o plata recaudada en el intervalo propuesto.

El siguiente TAD es una especificación para este problema.

**TAD** SAFA

**géneros** safa

**exporta** safa, generadores, observadores

**igualdad observacional**

$$(\forall s, s' : \text{safa}) \left( s =_{\text{obs}} s' \iff \begin{pmatrix} (\forall d{:}\text{nat})(\forall h{:}\text{nat}) \\ \#\text{Intervalo}(s,d,h) =_{\text{obs}} \#\text{Intervalo}(s',d,h) \\ \text{plataIntervalo}(s,d,h) =_{\text{obs}} \text{plataIntervalo}(s',d,h) \end{pmatrix} \wedge \right)$$

**observadores básicos**

| | | | | |
|---|---|---|---|---|
| #Intervalo | : | safa $\times$ nat $d$ $\times$ nat $h$ | $\longrightarrow$ | nat |
| plataIntervalo | : | safa $\times$ nat $d$ $\times$ nat $h$ | $\longrightarrow$ | nat |

**generadores**

| | | | | |
|---|---|---|---|---|
| iniciarDia | : | | $\longrightarrow$ | safa |
| agregarFactura | : | safa $\times$ nat $ts$ $\times$ nat $p$ | $\longrightarrow$ | safa $\{p > 0\}$ |

**axiomas**

| | | |
|---|---|---|
| #Intervalo(iniciarDia,$d$,$h$) | $\equiv$ | 0 |
| #Intervalo(agregarFactura$(s,t,p)$,$d$,$h$) | $\equiv$ | #Intervalo$(s,d,h)$ + **if** $d \leq t \wedge t < h$ **then** 1 **else** 0 **fi** |
| plataIntervalo(iniciarDia,$d$,$h$) | $\equiv$ | 0 |
| plataIntervalo(agregarFactura$(s,t,p)$,$d$,$h$) | $\equiv$ | plataIntervalo$(p,d,h)$ + **if** $d \leq t \wedge t < h$ **then** $p$ **else** 0 **fi** |

**Fin TAD**

Se pide dar una estructura de representación que permita realizar la operación *agregarFactura* en $O(n)$, e *#Intervalo* en $O(\log n)$, dónde $n$ es la cantidad de facturas agregadas hasta ese momento.

a) Describir la estructura a utilizar.

b) Escribir el invariante de representación de la estructura, tanto en castellano como en lógica.

c) Para ambas operaciones, explicar detalladamente como se usa la estructura de manera de cumplir con la complejidad pedida (no hace falta dar pseudocódigo de cada operación, pero se puede darlo si resulta conveniente para la explicación).

d) Agregar lo que sea necesario para proveer la operación *plataIntervalo* también en $O(\log n)$ e impleméntela. Además, describa las modificaciones a la estructura y a las operaciones del punto anterior.

**Pista:** Para *plataIntervalo*, notar que la cantidad de plata juntada entre los momentos $d$ y $h$ es igual a la cantidad juntada entre 0 y $h$ menos la cantidad juntada entre 0 y $d$.

## Ej. 2. Ordenamiento

La empresa que usa el SAFA tiene un sector de distribución de pedidos en su planta. En ésta se tienen $n$ cintas transportadoras, cada una pudiendo transportar un peso máximo medido en kilogramos. Cada cinta transporta un sólo paquete a la vez. Cuando llega un contingente de $k$ (con $k < n$) paquetes a transportar (de los cuales sólo interesa el peso en kilogramos), estos se quieren tratar de ubicar uno en cada cinta transportadora, donde cada paquete respete el límite de capacidad de la cinta asignada.

Dado un arreglo C de longitud $n$ con la capacidad máxima de cada cinta y un arreglo P de longitud $k$ con el peso de cada paquete, se desea saber si las $n$ cintas pueden transportar los $k$ paquetes.

*Nota:* No se puede asumir una cota sobre las capacidades de las cintas ni los pesos de los paquetes.

a) Implementar la función puedoTransportar?(**in** $C$: arreglo(nat), **in** $P$: arreglo(nat)) $\rightarrow$ bool que determina si es posible realizar el transporte. La complejidad temporal debe ser $O(n + k \log n)$ en peor caso.

*Sugerencia:* Se puede resolver el problema en dos etapas; la primera de costo $O(n)$ y la segunda $O(k \log n)$.

b) Justificar **detalladamente** la complejidad del algoritmo propuesto.

## Ej. 3. Dividir y conquistar

Dado un heap izquierdista implementado sobre árbol binario, se quiere diseñar un algoritmo que utilice la técnica de dividir y conquistar para obtener el camino a recorrer desde la raíz hasta la última hoja[1].

Elegir una representación adecuada para el árbol binario y para devolver el camino. Asumir que se cuenta con una función que devuelve la cantidad de elementos de un árbol binario en $O(1)$.

a) Implemente la función para devolver el camino a la última hoja de un heap. Muestre qué partes del algoritmo implementan cada una de las partes de la técnica de dividir y conquistar.

b) Calcule y **justifique adecuadamente** la complejidad temporal en peor caso. La misma debe ser $O(\log n)$ donde $n$ es la cantidad e nodos de elementos del heap.

[1] La última hoja es el nodo de más a la derecha del úlimo nivel del heap.