Universidad Simón Bolívar
Dpto. de Computación y Tecnología de la Información
CI2616 – Algoritmos y Estructuras II
Abril-Julio 2004

Carnet: _____

Nombre: _____

Examen Parcial I

(35 puntos)

Antes de empezar, revise bien el examen, el cual consta de 4 (CUATRO) preguntas.

Pregunta 1	Pregunta 2	Pregunta 3	Pregunta 4	Total
9 puntos	4 puntos	12 puntos	10 puntos	35 puntos

(Espacio adicional)

Pregunta 1 — 9 puntos

Al estudiar el comportamiento de dos algoritmos recursivos, se observó que ambos dividen, para un problema de tamaño n, el trabajo restante en seis partes, cada una de tamaño $n/2$. La diferencia entre ambos algoritmos radica en que el costo asociado a dividir en partes el problema original, y producir un resultado a partir de los subproblemas resueltos, es cuadrático para el primero (esto es, proporcional a n^2) y cúbico para el segundo (esto es, proporcional a n^3).

Para saber el costo de estos dos algoritmos, Ud. debe calcular las cotas "ajustadas" (notación Θ) de las siguientes recurrencias, donde c y c' son constantes positivas:

- $T(n) = 6 \cdot T(n/2) + c \cdot n^2$,
- $T(n) = 6 \cdot T(n/2) + c' \cdot n^3$.

Las cotas resultantes deben ser halladas mediante el Teorema Maestro.

Pregunta 2 — 4 puntos

En relación con el texto de Barbara Liskov y John Guttag, "*Program Development in Java – Abstraction, Specification, and Object-Oriented Design*", discuta semejanzas y diferencias entre el estilo de especificación de tipos abstractos de datos (TADs) utilizado en el texto y el utilizado en las clases de este curso.

Incluya en su discusión los siguientes aspectos: estilo básico de especificación de TADs (algebraico vs. modelos), modelo de representación, invariante de representación, relación de acoplamiento, y operaciones.

Pregunta 3 — 12 puntos

En esta pregunta trabajaremos con un tipo abstracto de datos (TAD) "Conjunto con Orden", el cual corresponde simplemente a conjuntos de elementos de un tipo cualquiera T, con la restricción de que se debe disponer de una relación de orden total "$\preccurlyeq$" sobre T.

Veamos parte de una especificación de este TAD con *modelo abstracto*:

Especificación $\mathbb{A}$ de TAD $CjtoOrd\,(T)$

Modelo de Representación

const $MAX : int$
var $s : set(T)$

Invariante de Representación

$\#s \leqslant MAX$

Operaciones

$\vdots$

proc $insertar\,($ **in-out** $c : CjtoOrd\,;$ **in** $x : T\,)$
$\{$ **Pre:** $x \in c.s \ \vee \ \#c.s < c.MAX$ $\}$
$\{$ **Post:** $c.s = c_0.s \cup \{\, x \,\}$ $\}$

proc $pertenece\,($ **in** $c : CjtoOrd\,;$ **in** $x : T\,;$ **out** $b : boolean\,)$
$\{$ **Pre:** true $\}$
$\{$ **Post:** $b \equiv (x \in c.s)$ $\}$

proc $extremos\,($ **in** $c : CjtoOrd\,;$ **out** $min : T\,;$ **out** $max : T\,)$
$\{$ **Pre:** $c.s \neq \varnothing$ $\}$
$\{$ **Post:** $min \in c.s \ \wedge \ max \in c.s \ \wedge \ (\forall\, x \,:\, x \in c.s \,:\, min \preccurlyeq x \preccurlyeq max\,)$ $\}$

$\vdots$

Fin TAD

Para implementar este TAD, se decide utilizar un arreglo como estructura de datos. A continuación se presenta parcialmente una especificación del TAD con este *modelo concreto* propuesto:

Especificación $\mathbb{C}$ de TAD $CjtoOrd\,(T)$, **refinamiento de** $\mathbb{A}$

Modelo de Representación

const $MAX : int$
var $elems : \mathbf{array}\,[\,0\,..\,MAX\,)\ \mathbf{of}\ T$
$tam : int$

Invariante de Representación

$0 \leqslant tam \leqslant MAX \quad \wedge \quad \ldots$

Relación de Acoplamiento

$s\ =\ \{\ i\ :\ 0 \leqslant i < tam\ :\ elems[i]\ \}$

Operaciones

$\vdots$

proc $insertar\,(\,\mathbf{in\text{-}out}\ c : CjtoOrd\,;\ \mathbf{in}\ x : T\,)$
{ **Pre:** ... }
{ **Post:** ... }

proc $pertenece\,(\,\mathbf{in}\ c : CjtoOrd\,;\ \mathbf{in}\ x : T\,;\ \mathbf{out}\ b : boolean\,)$
{ **Pre:** ... }
{ **Post:** ... }

proc $extremos\,(\,\mathbf{in}\ c : CjtoOrd\,;\ \mathbf{out}\ min : T\,;\ \mathbf{out}\ max : T\,)$
{ **Pre:** ... }
{ **Post:** ... }

$\vdots$

Fin TAD

Con toda esta información, responda ahora las siguientes preguntas:

3.1 (4 puntos) Indique qué se le debe agregar al invariante de representación de la especificación $\mathbb{C}$ con modelo concreto, a efectos de lograr que la implementación de la operación *pertenece* tome tiempo de ejecución $O(\lg n)$, siendo n el tamaño del conjunto de entrada c (esto es, $\#c.s$ en el modelo abstracto y $c.tam$ en el modelo concreto).

Explique brevemente el por qué de su respuesta.

3.2 (4 puntos) En la especificación $\mathbb{C}$ con modelo concreto, y según su respuesta a la pregunta **3.1**, ¿de qué orden de complejidad estima Ud. que será el tiempo de ejecución de la operación *insertar*?

Explique brevemente el por qué de su respuesta.

3.3 (4 puntos) En la especificación $\mathbb{C}$ con modelo concreto, complete la precondición y la postcondición de la operación *pertenece*.

(Espacio adicional para su respuesta a la pregunta 3)

Pregunta 4 — 10 puntos

En esta pregunta trabajaremos con un tipo abstracto de datos (TAD) Multi-Diccionario, correspondiente a una versión más flexible del conocido TAD Diccionario. La diferencia radica en que, mientras en un diccionario cada clave puede tener asociado a lo sumo un valor, en un multi-diccionario podremos tener muchos valores asociados a una clave.

En la especificación del TAD Multi-Diccionario que presentaremos a continuación, utilizaremos un modelo abstracto análogo al que hemos utilizado antes para diccionarios: un conjunto de claves conocidas; y una función de asociación de claves a valores, que ahora será multi-valuada, lo cual quiere decir que a cada clave se le asocia un conjunto de valores. En cuanto a la capacidad máxima de un multi-diccionario, ésta estará limitada por la cantidad total de asociaciones individuales clave-valor conocidas.

Restringiremos nuestra atención sólo a las siguientes operaciones del TAD: (i) *agregar*, que permite agregar una asociación clave-valor cualquiera al multi-diccionario; (ii) *eliminar*, que permite eliminar una asociación clave-valor pre-existente del multi-diccionario; (iii) *contarValores*, que cuenta los valores asociados a una clave dada; y (iv) *contarClaves*, que cuenta las claves asociadas a un valor dado.

Tal como el conocido TAD Diccionario, nuestro nuevo TAD está parametrizado por los tipos $T0$ y $T1$, correspondientes a los tipos de las claves y de los valores, respectivamente. Veamos entonces la especificación con *modelo abstracto*:

Especificación $\mathbb{A}$ de TAD *MultiDicc* $(T0, T1)$

Modelo de Representación

const $MAX : int$
var $conoc : set(T0)$
$tabla : T0 \nrightarrow set(T1)$

Invariante de Representación

$MAX > 0 \quad \wedge \quad conoc = \mathrm{dom}(tabla)$
$\wedge \quad (\forall\, x \,:\, x \in conoc \,:\, tabla(x) \neq \varnothing\,)$
$\wedge \quad (\sum x \,:\, x \in conoc \,:\, \#\, tabla(x)\,) \leqslant MAX$

Operaciones

$\vdots$

proc *agregar* (**in-out** $d : MultiDicc$; **in** $c : T0$; **in** $v : T1$)
{ **Pre:** $(\sum x \,:\, x \in d.conoc \,:\, \#\, d.tabla(x)\,) < d.MAX$ }
{ **Post:** ... }

proc *eliminar* (**in-out** $d : MultiDicc$; **in** $c : T0$; **in** $v : T1$)
{ **Pre:** $c \in d.conoc \;\wedge\; v \in d.tabla(c)$ }
{ **Post:** ... }

proc *contarValores* (**in** $d : MultiDicc$; **in** $c : T0$; **out** $n : int$)
{ **Pre:** true }
{ **Post:** $(c \notin d.conoc \;\Rightarrow\; n = 0\,) \;\wedge\; (c \in d.conoc \;\Rightarrow\; n = \#\, d.tabla(c)\,)$ }

proc *contarClaves* (**in** $d : MultiDicc$; **in** $v : T1$; **out** $n : int$)
{ **Pre:** true }
{ **Post:** $n = (\sum x \,:\, x \in d.conoc \wedge v \in d.tabla(x) \,:\, 1\,)$ }

$\vdots$

Fin TAD

Como estructura de datos para el TAD Multi-Diccionario se sugiere un arreglo de pares, con cada par representado por un registro (*record*). Cada uno de estos pares representará una asociación clave-valor de un multi-diccionario. Por ejemplo, si una clave a tuviese 2 valores asociados k y l, esa clave aparecería en 2 pares (a, k) y (a, l), una vez por cada valor.

Veamos la nueva especificación del TAD con este <u>*modelo concreto*</u> propuesto:

Especificación $\mathbb{C}$ de TAD *MultiDicc* $(T0, T1)$, **refinamiento de** $\mathbb{A}$

Modelo de Representación

const $MAX : int$
var ap : **array** $[\,0\,..\,MAX\,)$ **of record** $clave : T0\,;\ valor : T1$ **end**
$tam : int$

Invariante de Representación

$MAX > 0$
$\quad \wedge \quad 0 \leqslant tam \leqslant MAX$
$\quad \wedge \quad (\forall\, i, j \ :\ 0 \leqslant i, j < tam \ :\ ap[i] = ap[j] \ \Rightarrow\ i = j\ )$

Relación de Acoplamiento

$conoc \ = \ \{\ i \ :\ 0 \leqslant i < tam \ :\ ap[i].clave\ \}$
$\quad \wedge$
$tabla \ = \ \{\ x \ :\ x \in conoc \ :\ (\ x\ ,\ \{\ i \ :\ 0 \leqslant i < tam \ \wedge\ ap[i].clave = x \ :\ ap[i].valor\ \}\ )\ \}$

Operaciones

$\vdots$

proc *agregar* (**in-out** $d : MultiDicc\,;$ **in** $c : T0\,;$ **in** $v : T1\,$)
$\{$ **Pre:** $\ d.tam < d.MAX\ \}$
$\{$ **Post:** $\ \ldots\ \}$

proc *eliminar* (**in-out** $d : MultiDicc\,;$ **in** $c : T0\,;$ **in** $v : T1\,$)
$\{$ **Pre:** $\ (\exists\, i \ :\ 0 \leqslant i < d.tam \ :\ d.ap[i].clave = c \ \wedge\ d.ap[i].valor = v\ )\ \}$
$\{$ **Post:** $\ \ldots\ \}$

proc *contarValores* (**in** $d : MultiDicc\,;$ **in** $c : T0\,;$ **out** $n : int\,$)
$\{$ **Pre:** $\ \text{true}\ \}$
$\{$ **Post:** $\ n \ = \ (\sum i \ :\ 0 \leqslant i < d.tam \ \wedge\ d.ap[i].clave = c \ :\ 1\ )\ \}$

proc *contarClaves* (**in** $d : MultiDicc\,;$ **in** $v : T1\,;$ **out** $n : int\,$)
$\{$ **Pre:** $\ \text{true}\ \}$
$\{$ **Post:** $\ n \ = \ (\sum i \ :\ 0 \leqslant i < d.tam \ \wedge\ d.ap[i].valor = v \ :\ 1\ )\ \}$

$\vdots$

Fin TAD

Con toda esta información, responda ahora las siguientes preguntas:

4.1 (5 puntos) En la especificación $\mathbb{A}$ con modelo abstracto, complete la postcondición de la operación *agregar*. Se le sugiere utilizar análisis por casos, según si la clave de entrada era conocida previamente por el multi-diccionario o no.

4.2 (5 puntos) Demuestre formalmente que las precondiciones de la operación *eliminar* en las dos especificaciones se corresponden adecuadamente, según la noción de refinamiento de datos.

Para esta demostración se recomienda reescribir la segunda proposición de la relación de acoplamiento, notando que ésta implica la siguiente formulación en términos de aplicación de funciones:

$$(\forall\, x \,:\, x \in conoc \,:\, tabla(x) = \{\, i \,:\, 0 \leqslant i < tam \;\wedge\; ap[i].clave = x \,:\, ap[i].valor \,\})$$

(Espacio adicional para su respuesta a la pregunta 4)

(Espacio adicional para su respuesta a la pregunta 4)