Algoritmos y Estructuras de Datos II - 16 de abril de 2012

Parcial 1: Análisis de Algoritmos

Docentes: Daniel Fridlender y Silvia Pelozo

Alumno: ....

1. Dado el siguiente procedimiento:

$$
\begin{array}{l}
\textbf{proc}\ p\ (\textbf{in}\ /\ \textbf{out}\ \text{a} : \textbf{array}[1..N]\ \ \textbf{of int})\\
\quad \textbf{for}\ \ j := 1\ \ \textbf{to}\ \ N-1\ \ \textbf{do}\\
\quad\quad \textbf{for}\ \ k := j+1\ \ \textbf{to}\ \ N\ \ \textbf{do}\\
\quad\quad\quad \textbf{if}\ \text{a}[k] < \text{a}[j]\ \ \textbf{then}\\
\quad\quad\quad\quad \textit{swap}\ (\text{a},\ j,\ k)\\
\quad\quad\quad \textbf{fi}\\
\quad\quad \textbf{od}\\
\quad \textbf{od}
\end{array}
$$

a) Describir su objetivo.

b) Determinar la cantidad de comparaciones entre elementos de `a` que se realizan en el peor caso, en función de `N`.

c) ¿Qué característica (independiente del tamaño `N`) tiene el arreglo `a` en el peor caso? ¿Y en el mejor caso?

d) En el peor caso la cantidad de comparaciones entre elementos de `a` y la cantidad de intercambios es la misma. ¿Es apropiado considerar la operación de intercambio como la operación principal del algoritmo? ¿Por qué?

2. ¿Cuáles de las siguientes afirmaciones son verdaderas y cuáles falsas? Justificar con claridad.

a) sea $t(n) = n - 1.000.000$, entonces $\forall^\infty n \in \mathbb{N}.t(n) > 0$.

b) sea $t(n) = \begin{cases} 1 & \text{si } n \text{ es impar} \\ 0 & \text{si } n \text{ es par} \end{cases}$, entonces $\forall^\infty n \in \mathbb{N}.t(n) = 1$ y $\forall^\infty n \in \mathbb{N}.t(n) = 0$.

c) sea $f(n) \in \mathcal{O}(g(n))$ entonces $g(n)^2 \in \Omega(f(n) * g(n))$.

d) $f(n) \in \Theta(g(n))$ si y sólo si $g(n) \in \Theta(f(n))$.

3. Ordenar las siguientes funciones en orden incremental de acuerdo a sus $\mathcal{O}$, utilizando igualdad o inclusión estricta según corresponda. Por ejemplo, $\mathcal{O}(n+1) = \mathcal{O}(n) \subset \mathcal{O}(n^n)$.

a) $3^{\log_2 n}$ *b*) $3^{\log_3 n}$ *c*) $\sqrt{n}$ *d*) n *e*) $\log_2 n$ *f*) $n^2 - n\log n$

Justificar con claridad utilizando las propiedades que se demostraron, evitando en lo posible la utilización de la regla del límite.

4. Sean K y L constantes, y p el siguiente procedimiento:

$$
\begin{array}{l}
\textbf{proc}\ p(\textbf{in}\ n : \textbf{nat})\\
\quad \textbf{if}\ n \le 1\ \textbf{then skip}\\
\quad \textbf{else}\\
\quad\quad \textbf{for}\ i := 1\ \textbf{to}\ K\ \textbf{do}\ p(n\ \textbf{div}\ L)\ \textbf{od}\\
\quad\quad \textbf{for}\ i := 1\ \textbf{to}\ n\ \textbf{do}\\
\quad\quad\quad \textbf{for}\ j := 1\ \textbf{to}\ n\ \textbf{do}\ \textit{operación\_de\_}\mathcal{O}(1)\ \textbf{od}\\
\quad\quad \textbf{od}
\end{array}
$$

Determinar posibles valores de K y L de manera que el procedimiento tenga orden:

a) $\Theta(n^2 \log n)$ *b*) $\Theta(n^2)$ *c*) $\Theta(n^3)$

Justificar con claridad.

5. La siguiente recurrencia representa la cantidad de operaciones de un algoritmo en función del tamaño n de la entrada. Determinar la complejidad del algoritmo.

$$
t(n) = \begin{cases} n & \text{si } 0 \le n \le 1 \\ 2t(n-1) - t(n-2) + n & \text{si } n \ge 2 \end{cases}
$$