PONTIFICIA UNIVERSIDAD CATOLICA DE CHILE
ESCUELA DE INGENIERIA
DEPARTAMENTO DE CIENCIA DE LA COMPUTACION

Diseño y Análisis de Algoritmos - IIC2283
Tarea 1
Entrega: Viernes 15 de septiembre

1. [1.5 puntos] Una función $f : \mathbb{N} \to \mathbb{N}$ se dice no decreciente si $f(n) \leq f(n+1)$ para cada $n \in \mathbb{N}$. Demuestre o de un contraejemplo para la siguiente afirmación:

   - Dadas dos funciones $f, g : \mathbb{N} \to \mathbb{N}$ no decrecientes, si $f \in \Theta(g)$, entonces $\lim_{n\to\infty} \frac{f(n)}{g(n)}$ existe y es igual a un número $\ell \in \mathbb{R}^+$.

2. [1.5 puntos] Sea $T : \mathbb{N} \to \mathbb{N}$ una función dada por la siguiente ecuación de recurrencia:

$$T(n) \quad = \quad \begin{cases} n & n \leq 3 \\ T(\lfloor \frac{n}{2} \rfloor) + T(\lceil \frac{n}{2} \rceil) + T(\lceil \frac{n}{2} \rceil + 1) + n & n > 3 \end{cases}$$

   En esta pregunta usted debe encontrar una constante $k \in \mathbb{R}^+$ tal que $T \in \Theta(n^k)$, y debe demostrar que esta propiedad se cumple.

3. [1.5 puntos] Considere el algoritmo **InsertionSort** estudiado en clases, y suponga que la entrada de este algoritmo es una lista de números enteros sin repeticiones.

   En esta pregunta usted debe encontrar una función $f : \mathbb{N} \to \mathbb{N}$ tal que **InsertionSort** en el caso promedio es $\Theta(f(n))$, y debe demostrar que esta propiedad se cumple.

4. [1.5 puntos] Dado $n \in \mathbb{N}$, defina $\mathcal{F}_n$ como el conjunto de todas las funciones $f : \{0,1\}^n \to \{0,1\}$, y diga que $f \in \mathcal{F}_n$ es satisfacible si existe $w \in \{0,1\}^n$ tal que $f(w) = 1$. Además, defina el conjunto $\mathcal{F}$ de funciones Booleanas como $\bigcup_{n\in\mathbb{N}} \mathcal{F}_n$.

   Considere el siguiente algoritmo para verificar si una función Booleana es satisfacible.

   VerificarSat($f \in \mathcal{F}$)
   sea $n \in \mathbb{N}$ tal que $f \in \mathcal{F}_n$
   sea w el string 0^n
   while $w \neq 1^n$ **do**
   if $f(w) = 1$ **then return** sí
   else $w :=$ sucesor de w en orden lexicográfico
   if $f(w) = 1$ **then return** sí
   else return no

   En esta pregunta usted debe analizar la complejidad del algoritmo **VerificarSat** considerando como la operación a contar el número de accesos a la función f (el número de veces que realizamos llamadas de la forma $f(w)$). Suponiendo que cada acceso a f tiene costo uno, demuestre que **VerificarSat** en el caso promedio es $O(n)$.