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

Diseño y Análisis de Algoritmos - IIC2283
Tarea 1
Entrega: Viernes 9 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 todo $n \in \mathbb{N}$, y se dice estrictamente creciente si $f(n) < f(n+1)$ para todo $n \in \mathbb{N}$.

   Construya una función $f : \mathbb{N} \to \mathbb{N}$ tal que f es no decreciente, $f \in O(n^k)$ para algún $k \in \mathbb{N}$ y la siguiente condición **no** se cumple:

   $$(\exists c \in \mathbb{R}^+)(\exists n_0 \in \mathbb{N})(\forall n \geq n_0)(f(2 \cdot n) \leq c \cdot f(n)).$$

   Vale decir, para esta función f no es cierto que para n suficientemente grande se tiene que $f(2 \cdot n) \leq c \cdot f(n)$ para alguna constante c.

   Importante: En esta pregunta usted debe demostrar que la función f satisface las condiciones pedidas. Además, debe considerar que si la función que construye es estrictamente creciente entonces obtendrá un punto adicional.

2. [1.5 puntos] En clases analizamos la complejidad del algoritmo de multiplicación de Karatsuba realizando algunos supuestos sobre el peor caso. En esta pregunta usted debe resolver los siguiente problemas **sin** estos supuestos.

   a) [0.5 puntos] Sea $T(n)$ la cantidad de operaciones realizadas por el algoritmo de Karatsuba al multiplicar dos números con n dígitos en el peor caso. Construya una ecuación de recurrencia para $T(n)$ dando una justificación para los términos incluidos.

   b) [1 punto] Demuestre que $T(n) \in O(n^{\log_2(3)})$.

3. [1.5 puntos] Sea Σ un alfabeto y $\mathcal{A} : \Sigma^* \to \Sigma^*$ un algoritmo. Además suponga que para $n \in \mathbb{N}$ tenemos una distribución de probabilidad para las palabras en Σ^n, la cual esta dada por la función de probabilidad $\Pr_n$. Finalmente suponga que $\mathcal{A}$ funciona en tiempo polinomial en el caso promedio, vale decir, $\mathrm{E}(X_n) \in O(n^k)$ para algún $k \in \mathbb{N}$.

   Considere un algoritmo $\mathcal{B} : \Sigma^* \to \Sigma^*$, y para cada una de las siguientes afirmaciones de una demostración o un contraejemplo, suponiendo que la distribución de probabilidad usada para medir la complejidad en promedio de $\mathcal{B}$ está dada por $\Pr_n$.

   a) [0.4 puntos] Si para un polinomio fijo $p(n)$ se tiene que $\mathit{tiempo}_{\mathcal{B}}(w) = \mathit{tiempo}_{\mathcal{A}}(w) + p(|w|)$ para cada $w \in \Sigma^*$, entonces $\mathcal{B}$ funciona en tiempo polinomial en el caso promedio.

   b) [0.4 puntos] Si para un polinomio fijo $q(n)$ se tiene que $\mathit{tiempo}_{\mathcal{B}}(w) = \mathit{tiempo}_{\mathcal{A}}(w) \cdot q(|w|)$ para cada $w \in \Sigma^*$, entonces $\mathcal{B}$ funciona en tiempo polinomial en el caso promedio.

   c) [0.7 puntos] Si $\mathit{tiempo}_{\mathcal{B}}(w) = \mathit{tiempo}^2_{\mathcal{A}}(w)$ para cada $w \in \Sigma^*$, entonces $\mathcal{B}$ funciona en tiempo polinomial en el caso promedio.

4. [1.5 puntos] En esta pregunta usted debe modificar el procedimiento **Quicksort** estudiado en clases para que en el caso promedio sea $O(n \cdot \log_2(n))$ incluso si las listas a ordenar tienen elementos repetidos.

   En clases definimos **Quicksort** utilizando un procedimiento **Partición** tal que dada una lista L y un par (m, n) de valores con $m < n$, se tiene que la llamada **Partición**(L, m, n) retorna un valor i que satisface las siguientes condiciones: para cada $j \in \{m, \ldots, i-1\}$ se satisface que $L[j] \leq L[i]$, y para cada $j \in \{i+1, \ldots, n\}$ se satisface que $L[j] > L[i]$. Modifique este procedimiento para que dada una lista L (que puede tener elementos repetidos) y un par (m, n) de valores con $m < n$, se tiene que la llamada **Partición**(L, m, n) retorna un par (i_1, i_2) de valores tales que: (a) $i_1 \leq i_2$; (b) para cada $j \in \{m, \ldots, i_1 - 1\}$ se satisface que $L[j] < L[i_1]$; (c) para cada $j \in \{i_1, \ldots, i_2\}$ se satisface que $L[j] = L[i_1]$; y (d) para cada $j \in \{i_2 + 1, \ldots, n\}$ se satisface que $L[j] > L[i_1]$.

   Una vez hecha la modificación del procedimiento **Partición**, modifique la definición de **Quicksort** para que tome en cuenta el par de valores retornado por **Partición**, y demuestre que la nueva versión de **Quicksort** en el caso promedio es $O(n \cdot \log_2(n))$.