Algoritmos y Estructuras de Datos
Tarea 2

Profesores: José Galaviz y Sergio Rajsbaum
Ayudante: Armando Castañeda

fecha de hoy: 27 de agosto 2008, **fecha de entrega:** 10 de septiembre 2008
– No se aceptan tareas después de esta fecha
— *explica en detalle y con claridad <u>todas</u> tus respuestas* —
— *Tus algoritmos deberán ser lo más eficiente posibles* —
— *explica el funcionamiento de tus algoritmos informalmente, luego escribe el código, y luego demuestra correctez y complejidad.*—

Se permite trabajar en equipos de hasta DOS personas.
Pero cada uno debe entregar la tarea resuelta por separado, e indicar el nombre de su compañero de equipo.

Tema: Análisis de algoritmos, notación asintótica, gráficas y BFS

1. Escribe un resumen de no más de una cuartilla del capítulo de Dijkstra del las páginas 51 a la 67 del libro *Out of their Minds* de Shasha y Lazere.

2. Demuestra la correctez y analiza la complejidad del siguiente algoritmo de comparación lexicográfica. Este algoritmo compara cadenas en el sentido lexicográfico. Es decir, en el sentido en que lo hacen los diccionarios para ordenar palabras. **Entrada:** $c1$, $c2$: son cadenas de caracteres terminadas por el caracter especial de terminación de cadena '\0'. Este caracter, por definición, es menor que cualquier otro. **salida:** -1 si $c1$ es menor lexicograficamente que $c2$; 0 si $c1$ es igual a $c2$; 1 si $c1$ es mayor lexiograficamente que $c2$.

   Supondremos que las cadenas pueden indexarse para obtener sus caracteres individuales. Su primer índice es el 0. Es decir, para $C =$ "*hola*" tenemos $C[0] ==' h '$, $C[1] ==' o '$, $C[2] ==' l '$, $C[3] ==' a '$, $C[4] ==' \backslash 0 '$. Y sea $|C|$ la longitud de la cadena, en este caso, $|C| = 5$.

   Tip: Para la correctez observa cuales invariantes se cumplen justo antes de entrar al ciclo. Para la complejidad, considérala en función de $n = |c1| + |c2|$.

3. Para cada una de las siguientes funciones $f(n)$, determina el tamaño mas grande n de un problema que puede ser resuelto en tiempo t, suponiendo que el algoritmo resuelve el problema en $f(n)$ microsegundos. Funciones: n, $n!$, $(\log n)^2$, $n^{1/3}$, $n^{\log n}$, $n \log n$, $\log(n^2)$, n^2, 2^n, $n^{1/4}$, $\log n$. Tiempos t: 1 segundo, 1 minuto, 1 hora, 1 dia, 1 mes, 1 año, 1 siglo. Escribe una tabla con tus resultados, en la que cada columna está asociada una de las funciones, y ordenando las columnas de menor a mayor velocidad de crecimiento de las funciones (la 1era columna

```
Function compara_lexicograficamente(c1, c2):
        indice = 0
        do
            if c1[indice] < c2[indice] then return(−1) fi
            if c1[indice] > c2[indice] then return(1) fi
            if c1[indice] ==' \0 ' then return(0) fi
            indice = indice + 1
        while(TRUE)
```

Figure 1: Algoritmo de comparación lexicográfica.

con los datos de la funcion que crezca más despacio, la 2nda con los de la funcion que le siga en crecimiento, etc).

4. La notación $\log^a n$ se refiere a $(\log n)^a$. Demuestra para que valores de a, b se tiene que:

   (a) $log_a n \in \Theta(\log_b n)$

   (b) $\log^a n \in O(\log^b n)$

   (c) $\log^a n \notin \Omega(\log^b n)$

   (d) $n^a \in \Theta(n^3 + n^b + n \log n)$

5. Al ejecutar BFS sobre una gráfica conexa $G = (V, E)$, de $|V| = n$ vértices, la cola Q va teniendo tamaños $\ell_1, \ell_2, \cdots, \ell_n, \ell_{n+1}$, donde ℓ_i es igual al tamaño de Q al inicio de la i-ésima iteración del WHILE. Por lo tanto, $\ell_1 = 1$ y $\ell_{n+1} = 0$. ¿Qué secuencias son posibles? Para cada secuencia posible describe una gráfica y una ejecución de BFS que genere esa secuencia. Y para las secuencias que no son posibles, demuestra que no lo son.

6. Escribe una versión del algoritmo de BFS para gráficas dirigidas, demostrando su complejidad y correctez.

7. Da un ejemplo de una gráfica dirigida $G = (V, E)$, un vértice origen $s \in V$, y un conjunto de aristas de un árbol $E_\pi \subseteq E$ tal que para cada vértice $v \in V$ la (única) trayectoria en E_π de s a v es una trayectoria mínimia (lo mas corta posible) en G y aun así el conjunto E_π no puede ser generado corriendo BFS en G, no importa como se ordenen los vértices en la lista de adyacencias. ¿Esto puede suceder en una gráfica no dirigida? (explica)

Para cualquier duda contactar a Armando, José o a Sergio con suficiente anticipación.