Algoritmos y Estructuras de Datos
Tarea 3

Profesor: Sergio Rajsbaum
Ayudante: David Flores

fecha de hoy: 15 de septiembre 2005, **fecha de entrega:** 27 de septiembre 2005
– No se aceptan tareas después de esta fecha
— *explica en detalle y con claridad todas 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: invariantes y análisis de complejidad, notación asintótica, Euler

1. Escribe un resumen de no más de una cuartilla del capítulo de Rabin (página 68) del libro *Out of their Minds* de Shasha y Lazere.

2. Demuestra:

   (a) ¿Para qué valores de a, 2^{an} es $O(2^n)$?

   (b) ¿Para qué valores de a, b, a^n es $O(b^n)$?

   (c) ¿ Cómo se relacionan asintóticamente (notación O, Ω, Θ) $n!$ y 2^n ?

3. Considera una gráfica conexa cualquiera $G = (V, E)$. Este problema es acerca de diseñar un algoritmo para recorrer todas las aristas de G (no necesariamente sin repetir) partiendo del vértice s que eligas, y terminando en el vértice t que eligas. Supón que el costo de recorrer una arista es 1.

   (a) Presenta una familia infinita $G_1, G_2, \ldots$ de gráficas conexas, en la cual cada gráfica G_i tiene todos los vértices de grado impar. Para hacer esto considera el hipercubo en el que cada vértice es una palabra binaria de longitud k, y una arista conecta a dos vértices si y solo si sus palabras difieren en exactamente una posición.

   (b) Diseña un algoritmo de complejidad $O(m)$ que produzca un recorrido de costo a lo más $m + n/2 - 1$ sobre cualquier gráfica de la familia del inciso anterior de n vértices y m aristas. Además de analizar la complejidad y correctez de tu algoritmo, córrelo sobre la gráfica con $n = 8$ como ejemplo que ilustre sus propiedades.

4. Deriva la secuencia de Bruijn para $n = \sigma = 3$ a partir de la gráfica $G_{\sigma,n}$, explicando en detalle cada uno de los pasos de la derivación.

5. La siguiente función ordena el arreglo A de n elementos, todos ellos en el rango $1..m$, utilizando un arreglo auxiliar de m elementos, $cuantos[1..m]$, donde $m = 2^k$.

   (a) Escribe las invariantes para cada uno de los **for**'s.

   (b) Demuestra que la complejidad es $O(n + m)$.

   (c) ¿ Cuál es el tamaño de la entrada al algoritmo?

   (d) ¿Para qué valores de k el algoritmo es lineal?

   (e) Se sabe que la mejor complejidad de un algoritmo de ordenamiento por comparaciones es $\Theta(n \log n)$. En terminos de k, cuando conviene usar un algoritmo por comparaciones y cuando el counting_sort?

```
Function counting_sort(n, A, m):
        j = 1; k = 1;
        for i = 1 to m do cuantos[i] = 0 endfor
        for i = 1 to n do cuantos[A[i]] = cuantos[A[i]] + 1 endfor
        for i = 1 to m do
                for k = 1 to cuantos[i] do
                        A[j] = i;
                        j = j + 1;
                endfor
        endfor
```

Figure 1: Algoritmo de Ordenamiento por Conteo

Para cualquier duda contactar a David o a Sergio a colegadavid@gmail.com, rajsbaum@matem.unam.mx con suficiente anticipación.