UNIVERSIDAD DE COSTA RICA
FACULTAD DE INGENIERÍA
ESC. DE CS. DE LA COMP. E INFORMÁTICA

CI-1221 Estructuras de Datos y Análisis de Algoritmos
II ciclo 2010, grupo 03

I EXAMEN PARCIAL

Jueves 9 de julio

El examen consta de 6 preguntas que suman 126 puntos, pero no se reconocerán más de 110 (10 % extra). Cada pregunta empieza con un indicador de su puntaje y del tema tratado. Si la pregunta tiene subítemes, el puntaje de cada uno de ellos es indicado dentro de ellos. Se recomienda echar un vistazo a los temas de las preguntas y a su puntaje antes de resolver el examen, para así distribuir su tiempo y esfuerzo de la mejor manera. El examen se puede realizar con lápiz o lapicero. *No se permite el uso de calculadora.*

1. [10 pts.] *Matemática básica.*
   Muestre la propiedad de cambio de base:

$$\log_b a = \frac{\log_c a}{\log_c b}.$$

2. [23 pts.] *Análisis de la correctitud y del tiempo de ejecución de un algoritmo iterativo.*
   El siguiente algoritmo de ordenamiento se conoce como Ordenamiento de Burbuja:

```
Ordenamiento_de_Burbuja(A,n)
    for i = 1 to n-1
        for j = n downto i+1
            if A[j] < A[j-1]
                temp = A[j]
                A[j] = A[j-1]
                A[j-1] = temp
```

   a) Demuestre que el algoritmo es correcto. Para ello identifique el invariante del ciclo principal [6 pts.] y muestre cómo a partir de los pasos de inicialización [3 pts.], mantenimiento [6 pts.] y terminación [3 pts.] se infiere la correctitud del algoritmo.

   b) Determine el comportamiento asintótico del tiempo de ejecución del algoritmo. [5 pts.]

3. [33 pts.] *Solución de recurrencias.*
   Resuelva las siguientes recurrencias asumiendo que $T(n) = \Theta(1)$ para $n \leq 1$, $c > 0$ y $k > 1$.
   a) $T(n) = k\,T(n/k) + \sqrt{n}$. [5 pts.]
   b) $T(n) = 2T(n/4) + c\sqrt{n}$. [6 pts.]
   c) $T(n) = T(n-1) + c$. [7 pts.]
   d) $T(n) = T(n/2) + \lg n$. [15 pts.]

4. [17 pts.] *Análisis del tiempo de ejecución de un algoritmo recursivo.*
   El siguiente es el algoritmo de ordenamiento de Los 3 Chiflados:

   1. Si el último elemento es menor que el primer elemento, intercámbielos.
   2. Si el arreglo contiene 3 o más elementos:
      a) Llámese recursivamente para ordenar los primeros 2/3 del arreglo
      b) Llámese recursivamente para ordenar los últimos 2/3 del arreglo
      c) Llámese recursivamente para ordenar los primeros 2/3 del arreglo de nuevo

   a) Determine la complejidad computacional de este algoritmo [12 pts.] y compárela con la del mejor y peor algoritmos de ordenamiento estudiados en clase [3 pts.].
   b) Basado en el punto anterior, ¿le parece esta una forma inteligente de ordenar un arreglo o le parece completamente adecuado el nombre dado al algoritmo (o ninguna de las dos)? [2 pts.]

5. [28 pts.] *Simulación de ejecución de algoritmos.*
   Simule la ejecución de los siguientes algoritmos sobre el arreglo $A = \langle 3, 2, 1, 0 \rangle$, mostrando el (los) estado(s) del (los) (sub)arreglos después de cada "paso" (iteración del ciclo principal o llamado a [sub]función del algoritmo). Después del primer paso incorrecto el resto de pasos no suman puntos. [4 pts. c/ simulación, excepto ordenamiento por montículos: 4 pts. por monticularizar y 4 pts. por ordenar].
   a) Ordenamiento por selección.
   b) Ordenamiento por inserción.
   c) Ordenamiento por mezcla (*merge sort*).
   d) Ordenamiento rápido (*quicksort*, usando el primer o último elemento como pivote, **indique cuál**).
   e) Ordenamiento usando montículos (*heapsort*/)
   f) Ordenamiento por residuos (*radixsort*)

6. [15 pts.] *Ordenamiento por residuos (*Radixsort*).*
   Explique cómo se puede modificar el algoritmo de ordenamiento por residuos para ordenar arreglos que contengan tanto enteros positivos como negativos [8 pts.]. Concretice su idea utilizando seudocódigo [7 pts.].