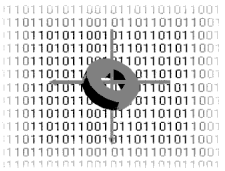

Universidad de Costa Rica
Facultad de Ingeniería
Esc. de Cc. de la Comp. e Informática

CI-1221 Estructuras de Datos y Análisis de Algoritmos
I ciclo 2012, grupo 02

I Examen Parcial

Martes 10 de abril

Nombre: _____ Carné: _____

El examen consta de 4 preguntas que suman 113 puntos, pero no se reconocerán más de 110 (10 % extra). Cada pregunta indica el tema tratado y su valor. Si la pregunta tiene subítemes, el puntaje de cada uno de ellos es indicado dentro de los subítemes. 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. Las preguntas se pueden responder en cualquier orden, pero se debe indicar en la tabla mostrada abajo el número de página del cuaderno de examen en la que *inicia* cada respuesta. Para ello las hojas del cuaderno de examen deben estar numeradas en la esquina superior externa de cada página. El examen se puede realizar con lápiz o lapicero. *No se permite el uso de dispositivos electrónicos* (calculadoras, teléfonos, audífonos, etc.).

Pregunta	Puntos	Páginas	Calificación
1. *Algoritmos iterativos: correctitud, duración y espacio*	20		
2. *Algoritmos iterativos y recursivos: duración*	20		
3. *Solución de recurrencias*	38		
4. *Simulación de ejecución de algoritmos*	35		
Total	113		

1. *Algoritmos iterativos: correctitud, duración y espacio.* [20 pts.]
   El siguiente algoritmo calcula la suma de los primeros n números naturales: $\sum_{k=1}^{n} k$.

   SUMA-DE-PRIMEROS-N-NATURALES(n)
   1 $S = 1$
   2 **para** $k = 2$ **hasta** n
   3 $\quad S = S + k$
   4 **devuelva** S

   a) Demuestre que el algoritmo es correcto. Para ello identifique un invariante apropiado para el ciclo [3 pts.] y muestre como los pasos de inicialización [2 pts.], mantenimiento [3 pts.] y terminación [2 pts.] permiten inferir la correctitud del algoritmo.

   b) Encuentre cotas asintóticas ajustadas para el tiempo de ejecución y espacio requeridos por el algoritmo. [1 pto c/u.]

   c) Escriba un algoritmo que calcule la suma de los primeros n números naturales en tiempo constante [3 pts.]. Demuestre la correctitud del algoritmo [5 pts.].

2. *Algoritmos iterativos y recursivos: duración.* [20 pts.]
   El siguiente algoritmo multiplica dos matrices cuadradas de tamaño $n \times n$ aplicando de forma directa la definición de multiplicación de matrices.

   MULTIPLICACIÓN-DE-MATRICES-CUADRADAS(A, B)
   1 Sea C la matriz resultante
   2 **para** $i = 1$ **hasta** n
   3 $\quad$ **para** $j = 1$ **hasta** n
   4 $\quad\quad c_{ij} = 0$
   5 $\quad\quad$ **para** $k = 1$ **hasta** n
   6 $\quad\quad\quad c_{ij} = c_{ij} + a_{ik} \cdot b_{kj}$
   7 **devuelva** C

   a) Determine una cota asintótica ajustada para el tiempo de ejecución del algoritmo. Explique. [7 pts.]

   En 1969 el Dr. Volker Strassen dio a conocer un método recursivo para multiplicar matrices cuadradas de tamaño $n \times n$ cuyo tiempo de ejecución cumple la siguiente recurrencia:

$$T(n) = \begin{cases} \Theta(1) & \text{si } n = 1, \\ 7\,T(n/2) + \Theta(n^2) & \text{si } n > 1. \end{cases}$$

   b) Determine una cota asintótica ajustada para el tiempo de ejecución del algoritmo. Muestre sus cálculos. [7 pts.]

   c) Compare sus respuestas en (a) y (b) y determine cuál algoritmo es más eficiente. [6 pts. Válidos solo si las respuestas en (a) y (b) son correctas].

3. *Solución de recurrencias.* [38 pts.]
   Resuelva las siguientes recurrencias asumiendo que $T(n) = \Theta(1)$ para $n \leq 1$ y $c > 0$.

   a) $T(n) = T(n/2) + c \lg (n^2) + c_0$. [10 pts.]

   b) $T(n) = 2\,T\left(\frac{n}{2\sqrt{2}}\right) + c\sqrt{n} + c_0$. [8 pts.]

   c) $T(n) = T\left(\frac{n-1}{2}\right) + c_0$. [20 pts.]

4. *Simulación de ejecución de algoritmos.* [35 pts.]
   Simule la ejecución de los siguientes algoritmos sobre el arreglo $A = \langle lu, ma, mi, ju, vi \rangle$, mostrando el (los) estado(s) del (los) (sub)arreglos después de cada "paso" (iteración del ciclo principal o llamado a subrutina del algoritmo). Asuma un criterio de ordenamiento *alfabético*. Después del primer paso incorrecto el resto de pasos no suman puntos. [5 pts. c/ simulación, excepto ordenamiento por montículos: 5 pts. por monticularizar y 5 pts. por ordenar].

   a) Ordenamiento por selección.

   b) Ordenamiento por inserción.

   c) Ordenamiento por mezcla [*mergesort*]. (Si el tamaño del [sub]arreglo es impar, *redondee* el punto medio hacia abajo, es decir, que la parte izquierda del subarreglo sea más pequeña que su parte derecha, y no al revés. Por ejemplo, un arreglo de tamaño 5 debería ser partido en un arreglo de tamaño 2 [a la izquierda] y uno de tamaño 3 [a la derecha]).

   d) Ordenamiento rápido [*quicksort*] (usando el último elemento como pivote).

   e) Ordenamiento usando montículos [*heapsort*]

   f) Ordenamiento por residuos [*radixsort*] (asumiendo que en cada pasada se toma un caracter distinto).