DECSAI
Departamento de Ciencias de la Computación e I.A.
Universidad de Granada

Teoría de Algoritmos
Segundo de Ingeniería Informática
Examen de Febrero del Curso 2004-2005

1. Definición formal de algoritmo. Características primordiales de un algoritmo. Concepto de caso de un problema. Definición de tamaño de un caso. Tipos de casos. En particular, considere un vector A de tamaño n, constituido por elementos de la misma naturaleza. Sea T(m) el tiempo medio que requiere una llamada al conocido algoritmo "Quicksort" para un sub-vector A[i..j], donde m = j-i+1 es el numero de elementos que hay en el correspondiente sub-vector. Escribir y justificar una ecuacion para calcular el tiempo medio requerido para ejecutar "Quicksort" en el vector A. ¿Que tiempo consumira la operacion de pivoteo que se supone realiza el algoritmo?.

2. La eficiencia de cierto algoritmo está dada por la siguiente ecuación

$$t_C(n) = t_A(n) \qquad \text{si } n \le n_0$$
$$= 3\,t_C(n/2) + t(n) \qquad \text{si } n \ge n_0$$

   donde n_0 es el umbral del problema, cuando se resuelve con la técnica Divide y Vencerás. Definir el papel que n_0 desempeñan en esa ecuación, y calcular su valor cuando A se implementa con un tiempo de ejecución de $t_A(n) = n^3$ segundos, y t(n) es otro tiempo de ejecución de valor t(n) = 530n segundos.

3. Supongamos que disponemos de *n* ficheros $f_1, f_2, ..., f_n$ con tamaños $l_1, l_2, ..., l_n$ y un disquete de capacidad $d < l_1 + l_2 + ... + l_n$.

   a. Queremos maximizar el número de ficheros que ha de contener el disquete, y para eso ordenamos los ficheros por orden creciente de su tamaño y vamos metiendo ficheros en el disco hasta que no podamos meter más. Determinar si este algoritmo encuentra solución óptima en todos los casos.

   b. Queremos llenar el disquete tanto como podamos, y para eso ordenamos los ficheros por orden decreciente de su tamaño, y vamos metiendo ficheros en el disco hasta que no podamos meter más. Determinar si este algoritmo encuentra solución óptima en todos los casos.

4. Explicar las fases en las que se aplica la técnica de diseño de Programación Dinámica, asi como las condiciones que han de darse para que un problema pueda resolverse con ese metodo. Explicar si es posible o no aplicar Programación Dinámica al diseño de un algoritmo para el cálculo de la sucesión de los números de Fibonacci.

5. Explicar la diferencia existente entre los métodos "Backtracking" y "Branch and Bound". Explicar como se aplican las estrategias LIFO, FIFO y LC a la lista de nodos vivos en la técnica "Branch and Bound", y si pueden o no aplicarse a la "Backtracking".

- **Tiempo para la realización del examen: 3 horas**
- **No está permitido el uso de apuntes, libros o cualquier otro material de consulta**