UNIVERSIDAD NACIONAL DE EDUCACIÓN A DISTANCIA	**Programación III** Solución **Prueba Presencial**	**Prueba Presencial** **2ª Semana** Febrero de 2010 Duración: **2 horas** Material permitido: **NINGUNO**

Cuestión 1 (2 punto). Dado el montículo de mínimos m=[2,6,2,12,7,6,9], añadir un 3 al montículo. Detallar todos los pasos. Escribir el montículo resultante.

Solución:

Se añade al final de m el valor 3, con m=[2,6,2,12,7,6,9,3] siendo i=8 la posición, se comprueba que m[4] sea mayor, y se intercambia quedando m=[2,6,2,3,7,6,9,12], se comprueba también con m[2] y se intercambia quedando m=[2,3,2,6,7,6,9,12] el montículo resultante

Cuestión 2 (2 puntos). Hallar y demostrar formalmente la relación de pertenencia entre f(n) y O(g(n)) y también entre g(n) y O(f(n)) considerando $f(n)= 2^n$ y $g(n)= 2^{2n}$.

Solución:

En el primer caso se trata de encontrar que existe *c* y existe n_0 tales que para todo $n \geq n_o$, $f(n) \leq c \cdot g(n)$. Tenemos que suponer que existe un n_0 tal que para todo $n \geq n_o$ $2^n \leq c \cdot 2^{2n}$ es decir (aplicando $\log_2$) que $n \leq c' + 2n$ para todo $n \geq n_o$ lo que es cierto para cualquier $n_0 \neq 0$.

Se puede comprobar que el contrario, el segundo caso, no es cierto, basta con considerar c=1 y $n_o=1$.

Para la demostración también se puede utilizar la regla del límite que puede ver en la página 96 del libro base de la asignatura:

$$\lim_{n \to \infty} f(n)/g(n) = \lim_{n \to \infty} 2^n/2^{2n} = \lim_{n \to \infty} 1/2^n = 0$$

con lo que $f(n) \in O(g(n))$ y $g(n) \notin O(f(n))$.

Cuestión 3 (2 puntos). Considera el problema de la mochila, en el que tenemos una mochila de capacidad *M*, *n* objetos con beneficios b_1, b_2, b_3, ... b_n y pesos p_1, p_2, p_3, ... p_n. El objetivo es maximizar el valor de los objetos transportados, respetando la limitación de la capacidad impuesta *M*. Indica qué esquema o esquemas consideras **más adecuados** para resolver este problema en los siguientes casos:

a) Cada objeto puede meterse en la mochila, no meterse, o meter la mitad del objeto obteniendo la mitad del beneficio.
b) Cada objeto puede meterse en la mochila o no meterse, pero no se puede fraccionar.

Además de nombrar el esquema o esquemas, explica el porqué de su elección, los aspectos destacados de cómo resolverías el problema (función de selección, restricciones, cotas…, en función del esquema propuesto) y el coste asociado. No se piden los algoritmos.

Solución:

Ambos casos, el a) y b), son similares, no existe una función de selección que garantice encontrar una solución óptima. Aunque el caso a) permite fraccionar los elementos a la mitad obteniendo la mitad del beneficio, esto no hace que cambie la naturaleza del problema, ya que simplemente supone que podríamos tratar el mismo problema como si hubiera otros *n* elementos más, que serían los correspondientes a mitad de peso y beneficio de los ya existentes. Para que el caso a) se pudiera solucionar con el esquema voraz se deberían poder partir los objetos en cualquier fracción.
Como es un problema de optimización y no se puede aplicar el esquema voraz, el esquema más apropiado es el de **Ramificación y Poda**.

En el árbol de exploración, en la raíz no estaría incluido ningún objeto y en cada nivel sucesivo se determinaría la inclusión de un objeto. De cada nodo sólo se generarán los sucesores que satisfagan la restricción de peso *M*.
Una función de coste que estima el valor de la solución si seguimos por ese camino podría ser el cálculo del beneficio de los objetos ya almacenados en la mochila, más el beneficio de objetos aun no estudiados, teniendo en cuenta la restricción de peso y seleccionándolos en el orden que se haya establecido, por ejemplo ordenados de mayor a menor valor por unidad de peso.
Como solución inicial o primera cota superior se podría aplicar un algoritmo voraz considerando que los objetos sí se pueden fraccionar, o seguir el mismo criterio que la función de estimación: coger objetos y meterlos en la mochila hasta que no se puedan meter más debido a la restricción de peso.
El coste de la solución viene dado principalmente por el número de nodos del espacio de búsqueda que en este caso es $O(2^n)$.

Problema (4 puntos). (Torres de Hanoi) Se tienen 3 palos verticales y n discos agujereados por el centro. Los discos son todos de diferente tamaño y en la posición inicial se tienen insertados en el primer palo ordenados en tamaños en sucesión decreciente desde la base hasta la altura. El problema consiste en pasar los discos del 1er al 3er palo (también ordenados de mayor a menor desde la base hasta la altura) observando las siguientes reglas: a) Se mueven los discos de 1 en 1. Y b) nunca un disco puede colocarse encima de uno menor que éste. Se pide:

1. Elección del esquema más apropiado, el esquema general y explicación de su aplicación al problema (0,5 puntos).
2. Descripción de las estructuras de datos necesarias (0.5 puntos).
3. Algoritmo completo a partir del refinamiento del esquema general (2,5 puntos).
4. Estudio del coste del algoritmo desarrollado (0.5 puntos).

Solución:

Indicaciones:

1. El esquema más apropiado es **divide y vencerás**.
2. No es necesaria ninguna estructura de datos compleja si se muestran los movimientos, por ejemplo:"el disco n va al palo x". Sólo hacen falta tipos de datos estándar.

4 Coste: $O(2^n)$

La solución completa está en:
Gonzalo, J.; Rodríguez, M. *Esquemas algorítmicos: enfoque metodológico y problemas resueltos*, UNED, 1997.