

Apellido y Nombre: \_\_\_\_\_

Carrera: \_\_\_\_\_ DNI: \_\_\_\_\_

[Llenar con letra mayúscula de imprenta GRANDE]

## Algoritmos y Estructuras de Datos. Final. [7 de Julio de 2005]

### [Ej. 1] [clases (30 pts)]

- a) [aoo (30 pts)] Escribir los siguientes métodos del **TAD árbol ordenado orientado**: `insert(p,x)`, `erase(p)`, `splice(to,from)`.

### [Ej. 2] [programacion (46 pts)]

- a) [parc-ord (34 pts)] Recordemos que un árbol es parcialmente ordenado si dados dos nodos  $m$ ,  $n$  tal que  $m$  es hijo de  $n$ , entonces el valor de  $m$  es mayor o igual al de  $n$ .  
*Consigna:* Escribir un predicado  
`bool es_parcialmente_ordenado(tree<int> &T, bool (comp*)(int,int))`, que verifica si el árbol ordenado orientado  $T$  es parcialmente ordenado con respecto a la función de comparación `comp()`.
- b) [proper-subset (12 pts)] Dada una lista de árboles `list< set<int> > L`, escribir una función predicado `bool proper_subset(list< set<int> > &L)`, que determina si los conjuntos de la lista  $L$  son subconjuntos propios en forma consecutiva. Es decir, si  $L = (A_0, A_1, \dots, A_{n-1})$ , determinar si  $A_j \subset A_{j+1}$  para  $j = 0, \dots, n-2$ . (Recordar que  $A \subset B$  indica inclusión propia, es decir  $A \subseteq B$  y  $A \neq B$ .)

### [Ej. 3] [operativos (16 pts)]

- [t-exec (4 pts)]  
Dadas las funciones
  - $T_1(n) = 3n^2 + 2n^3 + 0.5 \cdot 3^n$ ,
  - $T_2(n) = 4^5 + 10n + 0.2n \log_2 n$ ,
  - $T_3(n) = 3n + 5 + 6\sqrt{n}$ ,
  - $T_4(n) = n^{1.5} + 7! + 4\sqrt{n}$ .ordenarlas de menor a mayor.  
 $T_{\square} < T_{\square} < T_{\square} < T_{\square}$
- [huffmann (4 pts)] Dados los caracteres siguientes con sus correspondientes probabilidades, contruir el código binario y encodar la palabra **SKYWALKER**:  $P(S) = 0.1, P(K) = 0.2, P(Y) = 0.05, P(W) = 0.05, P(A) = 0.05, P(L) = 0.05, P(E) = 0.2, P(R) = 0.3$  Calcular la longitud promedio del código obtenido.
- [heap-sort (4 pts)] Dados los enteros  $\{23, 15, 12, 6, 8, 13, 10, 9\}$  ordenarlos por el método de “montículos” (“heap-sort”). Mostrar el montículo (minimal) antes y después de **cada** inserción/supresión.
- [quick-sort (4 pts)] Dados los enteros  $\{3, 1, 11, 14, 18, 17, 7, 2, 11, 20\}$  ordenarlos por el método de “clasificación rápida” (“quick-sort”). En cada iteración indicar el pivote y mostrar el resultado de la partición.

### [Ej. 4] [preguntas (8 pts, 2 por pregunta)]

- a) ¿Cómo es el tiempo de ejecución para intercalar dos listas clasificadas de  $n$  elementos?
- ☐ ...  $O(1)$
- ☐ ...  $O(n)$

Apellido y Nombre: \_\_\_\_\_

Carrera: \_\_\_\_\_ DNI: \_\_\_\_\_

[Llenar con letra mayúscula de imprenta GRANDE]

- ☐ ...  $O(n^2)$   
☐ ...  $O(\log n)$
- b) ¿Cuál de los siguientes árboles es un árbol binario de búsqueda?
- ☐ ... (6 (5 (3 . 4 ) .) (9 8 10))  
☐ ... (6 (5 (3 . 4 ) .) (9 (8 2 .) 10))  
☐ ... (6 (5 (3 4 .) .) (9 8 10))  
☐ ... (6 (5 (3 . 4 ) 7) (9 8 10))
- c) ¿Cuál es el tiempo de ejecución para el algoritmo de ordenamiento por montículos en el peor caso?
- ☐ ...  $O(n)$   
☐ ...  $O(\log n)$   
☐ ...  $O(1)$   
☐ ...  $O(n \log n)$
- d) ¿Cuál es el tiempo de ejecución para la función `push()` del TAD COLA, implementado sobre listas (asumiendo que el frente de la cola es el principio de la lista)?
- ☐ ...  $O(n)$   
☐ ...  $O(\log n)$   
☐ ...  $O(1)$   
☐ ...  $O(n \log n)$