Apellido y Nombre: ____________________

Carrera: ________ DNI: ____________________

[Llenar con letra mayúscula de imprenta GRANDE]

Universidad Nacional del Litoral
Facultad de Ingeniería y Ciencias Hídricas
Departamento de Informática

Algoritmos y Estructuras de Datos

Algoritmos y Estructuras de Datos. Examen Final. [05 de Mayo de 2005]

[Ej. 1] [Clases (30 puntos)]

Escribir los siguientes métodos del TAD `btree`: `insert(p,x)`, `erase(p)`, `find(x)`. Escribir las declaraciones de la clase y los componentes necesarios para implementar las funciones indicadas.

[Ej. 2] [Programación (total = 70 puntos)]

a) **[incluido (45 puntos)]**
Escribir un predicado `bool incluido(tree<int>&A,tree<int>&A);`, el cual retorna verdadero si la *estructura* del árbol del nodo `B` está *"incluida"* dentro del árbol `A`, independientemente de las etiquetas de los nodos correspondientes. Por ejemplo, si tenemos los árboles `T1=(z q r (t u v))`, `T2=(x y (p w s))`, `T3=(a c d (e j))`, entonces tenemos que `T3` está incluido en `T1`, pero `T2` no está incluido en `T1`.

b) **[separa (25 puntos)]**
Escribir una función `void separa(queue<int>&Q, queue<int>&Qt,queue<int>&Qf,bool (*pre)(int));` que separa los valores de la cola `Q` en dos colas `Qf`, `Qt`, tal que los valores que satisfacen el predicado `pred()` van a la cola `Qt` mientras que los que no lo satisfacen van a la cola `Qf`. Por ejemplo, si `Q={1,3,2,4,3,2,5}` y el predicado es `bool impar(x) {return x %2; }`, entonces después de `separa(Q,Qt,Qf)` debe quedar `Qt={1,3,3,5}`, `Qf={2,4,2}`.
Restricciones: El algoritmo debe tener un tiempo de ejecución $O(n)$, donde n es el número de elementos en la cola original. No se deben usar estructuras auxiliares. El algoritmo debe ser *"estable"*.

[Ej. 3] [LIBRES - operativos (total = 80 puntos)]

a) **[rec-arbol (20 pt)]** Dibujar el árbol ordenado orientado cuyos nodos, listados en orden previo y posterior son

- `ORD_PRE` $=\{Z, A, B, C, J, M, Q, N, P, K, D\}$,
- `ORD_POST` $=\{A, Q, M, P, N, J, K, C, D, B, Z\}$.

b) **[huffman (20 pt)]** Dados los caracteres siguientes con sus correspondientes probabilidades, contruir el código binario y encodar la palabra `BENEDICTO` $P(B) = 0.2, P(O) = 0.2, P(N) = 0.2, P(E) = 0.1, P(D) = 0.1, P(I) = 0.1, P(C) = 0.05, P(T) = 0.05$ Calcular la longitud promedio del código obtenido.

c) **[misc-arbol (20 pt)]:** Dado el árbol `(z q (r t s) p)`,

1) Cuál es el nodo que está a la vez a la izquierda de `p` y a la derecha de `q` y es antecesor propio de `t`?
2) Particione el árbol con respecto al nodo `t`, es decir indique cuales son sus antecesores y descendientes propios, derecha e izquierda

d) **[heap-sort (20 pt)]** Dados los enteros $\{12, 11, 14, 8, 16, 3, 9\}$ ordenarlos por el método de *"montículos"* (*"heap-sort"*). Mostrar el montículo (minimal) antes y después de cada inserción/supresión.

[Ej. 4] [LIBRES - preguntas (total = 20 pt, 5pt/preg)] Responder según el sistema *"multiple choice"*, es decir marcar con una cruz el casillero apropiado. **Atención:** Algunas respuestas son intencionalmente "descabelladas" y tienen puntajes **negativos**!!]

a) Dadas las funciones $T_1(n) = 0.5n + \sqrt{n}$, $T_2(n) = 0.2n^2 + 2.1 \log n$, $T_3(n) = 3n! + 5.4n^3$ y $T_4(n) = n^{1/2}$ decir cuál de los siguientes ordenamientos es el correcto

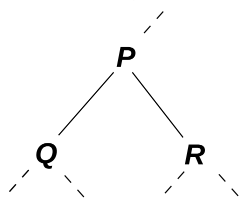
☐ $T_4 < T_3 < T_2 < T_1$
☐ $T_4 < T_1 < T_2 < T_3$
☐ $T_2 < T_1 < T_4 < T_3$
☐ $T_3 < T_4 < T_1 < T_2$

b) El tiempo de ejecución para el algoritmo de clasificación por montículos (*"heapsort"*) es $O(n \log n)$ (n es el número de elementos a ordenar) ...

☐ ... siempre.
☐ ... a veces.
☐ ... nunca.
☐ ... en el mejor caso.

c) El montículo es un árbol binario que satisface la condición de ser *"parcialmente ordenado"*. Si P, Q, R son las etiquetas del nodo y sus dos hijos, la condición de parcialmente ordenado se expresa como (*Nota: Consideramos un montículo "minimal"*):

☐ $Q + R \leq \infty$.
☐ $Q \leq P \leq R$.
☐ $P \leq 1/2(Q + R)$.
☐ $P \leq Q, R$.



d) ¿Cuál es el tiempo de ejecución del procedimiento de clasificación por *incrementos decrecientes* (*shell-sort*) en el caso promedio?

☐ $O(n^{1.3})$
☐ $O(n^{1.5})$
☐ $O(\log n)$
☐ $O(n!)$