

Algoritmos y Estructuras de Datos.

Examen Final. [6 de Octubre de 2005]

El examen se compone de dos partes.

- Clases y programación: TODOS.
- Operativos y preguntas: Sólo LIBRES.

Los libres deben cumplir con al menos un 70 % del porcentaje de la segunda parte.

Ej. 1.- [todos: queue (20 pt)] Escribir las funciones del TAD COLA (queue<>). Incluir el header y todas las declaraciones necesarias.

Ej. 2.- [todos: Ejercicios de programación]

- a) **[verifica-abb (50 pt)]** Escribir una función predicado
`bool verifica_abb(tree<int> &A, bool (*less)(int,int));` que verifica si el subárbol de un nodo `n` verifica la condición de *árbol binario de búsqueda* con respecto a la relación de orden `less()`.
- b) **[elimina-valor (30 puntos)]**
 Escribir una función `void elimina_valor(queue<int>&C, int);` que elimina todas las ocurrencias del valor `n` en la cola `C`. Por ejemplo, si `C = {1,3,5,4,2,3,7,3,5}`, después de `elimina_valor(C,3)` debe quedar `C = {1,5,4,2,7,5}`. *Sugerencia: Usar una estructura auxiliar lista o cola.*
Restricciones: El algoritmo debe tener un tiempo de ejecución $O(n)$, donde n es el número de elementos en la cola original.

Ej. 3.- [libres: Ejercicios operativos. (total 80pt)]

- a) **[particionar (20 pt)]** Considerando el árbol $(z \ q \ (r \ a \ (b \ 1)) \ (t \ n \ m))$, decir cuál son los nodos descendientes(`q`), antecesores(`q`), izquierda(`q`) y derecha(`q`).
- b) **[heap-sort (20 pt)]** Dados los enteros $\{6, 8, 11, 4, 5, 9, 6, 7, 10\}$ ordenarlos por el método de “montículos” (“*heap-sort*”). Mostrar el montículo (minimal) antes y después de **cada** inserción/supresión.
- c) **[huffman (20 ptos)]** Dados los caracteres siguientes con sus correspondientes probabilidades, construir el código binario y encodar la palabra KATHRINA
 $P(K) = 0,05, P(H) = 0,2, P(T) = 0,1, P(R) = 0,05, P(A) = 0,2, P(N) = 0,15, P(I) = 0,1, P(P) = 0,05, P(S) = 0,05, P(B) = 0,05$. Calcular la longitud promedio del código obtenido.
- d) **[rec-arbol (20 ptos)]** Dibujar el árbol ordenado orientado cuyos nodos, listados en orden previo y posterior son
 - $ORD_PRE = \{W, X, Y, Z, R, P, Q, T, V, U\}$,
 - $ORD_POST = \{X, Y, R, P, T, V, U, Q, Z, W\}$.

Ej. 4.- [libres: preguntas (20 pt)] 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!!**

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

a) ¿Cómo es el tiempo de ejecución para intercalar dos listas clasificadas de n elementos?

- ☐ ... $O(1)$
- ☐ ... $O(n)$
- ☐ ... $O(n^2)$
- ☐ ... $O(\log n)$

b) ¿Cuál de los siguientes árboles es un árbol binario de búsqueda?

- ☐ ... (7 5 (9 6 (10 . 12))))
- ☐ ... (7 5 (9 (8 6 .) (13 . 12))))
- ☐ ... (7 5 (9 8 (10 . 12))))
- ☐ ... (7 5 (9 8 (10 12))))

c) El tiempo de ejecución para el algoritmo de *clasificación rápida* (*quick-sort*) en el peor caso es:

- ☐ ... $O(n \log n)$
- ☐ ... $O(n^2)$
- ☐ ... $O(1)$
- ☐ ... $O(n)$

d) ¿Cuál es el tiempo de ejecución de la operación de *re-heap* en montículos?

- ☐ ... Caso promedio $O(\log n)$.
- ☐ ... Siempre $O(\log n)$.
- ☐ ... Caso promedio $O(n \log n)$.
- ☐ ... Siempre $O(n \log n)$.