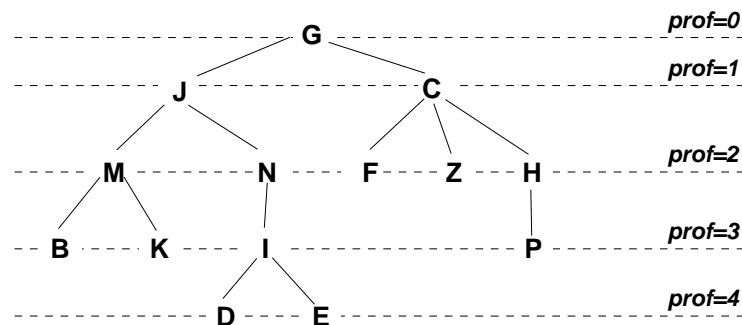


Algoritmos y Estructuras de Datos.

Examen Final. [13 de Febrero de 2003]

- Ej. 1.-** Escribir las funciones primitivas del TAD LISTA con celdas simplemente enlazadas por punteros ó cursores. Es decir, implementar en Pascal los siguientes procedimientos/funciones listadas abajo. Incluir todas las definiciones de tipo necesarias. INSERTA(x,p,L), LOCALIZA(x,L), RECUPERA(p,L), SUPRIME(p,L), SIGUIENTE(p,L), ANULA(L), PRIMERO(L), y FIN(L).
- Ej. 2.-** Escribir un función `function CUENTA_PROF(n:nodo; m:integer; A:arbol) : integer;` que dado un nodo **n** en un árbol **A** cuenta el número de nodos del subárbol de **A** cuya raíz es **n** que están a profundidad **m** o menor (con respecto a **n**). Por ejemplo, para el árbol de la figura debe retornar

CUENTA_PROF(G,2,G) -> 8
 CUENTA_PROF(J,1,G) -> 3
 CUENTA_PROF(N,3,G) -> 4



Usar las primitivas de árbol ordenado orientado siguientes:

HIJO_MAS_IZQ(n,A), HERMANO_DER(n,A). *Sugerencia:* Hacer la función *recursiva*. Notar que, por ejemplo:

$$\text{CUENTA_PROF}(G,2,G) = 1 + \text{CUENTA_PROF}(J,1,G) + \text{CUENTA_PROF}(C,1,G)$$

La recursividad de la función debe cortar cuando $n = \Lambda$ o $m < 0$.

Ej. 3.- Ejercicios básicos sobre TAD's

- (a) Escribir un procedimiento `procedure SACAPAR(var L:lista; C:cola);` que apendiza a la lista **L** todos los elementos de **C** que son pares, los cuales a su vez deben ser removidos de **C**. Se puede usar una estructura auxiliar (cola o lista). Por ejemplo, si inicialmente $L=\{2,3,4\}$ y $C=\{1,6,3,5,2,8\}$ entonces, después de hacer SACAPAR(L,C) debe quedar $L=\{2,3,4,6,2,8\}$ y $C=\{1,3,5\}$. Utilizar las siguientes primitivas:

- **TAD LISTA:** INSERTA(x,p,L), RECUPERA(p,L), SUPRIME(p,L), SIGUIENTE(p,L), ANULA(L), PRIMERO(L), y FIN(L).
- **TAD COLA:** ANULA(C), PONE_EN_COLA(x,C), QUITA_DE_COLA(C), VACIA(C), y FRENTE_DE_COLA(C)

- (b) Escribir un procedimiento `procedure ELIMINA_VALOR(var C:cola; n: integer);` 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.* El algoritmo debe tener un tiempo de ejecución $O(n)$, donde n es el número de elementos en la cola original. Utilizar las primitivas del **TAD COLA** listadas en el ejemplo anterior.

Ej. 4.- [LIBRES] Ejercicios operativos:

- (a) **Árboles:** Dibujar el árbol ordenado orientado cuyos nodos, listados en orden previo y posterior son
- $ORD_PRE = \{A, Z, S, T, U, W, Q, R, B\}$.
 - $ORD_POST = \{T, W, U, S, Z, R, B, Q, A\}$.
- (b) **[LIBRES]** Dados los enteros $\{9, 8, 11, 5, 13, 10, 6, 4, 12, 18, 1\}$ insertarlos, en ese orden, en un “árbol binario de búsqueda”. Mostrar las operaciones necesarias para eliminar los elementos 9, 6 y 10.

Ej. 5.- [LIBRES] Preguntas: [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 $T_1(n) = 5n + \log n$, $T_2(n) = 4n^2 + \sqrt{n}$, $T_3(n) = 2^n + n!$ y $T_4(n) = \sqrt{n} + \log n$ decir cuál de los siguientes ordenamientos es el correcto
- ☐ $T_3 < T_4 < T_1 < T_2$
- ☐ $T_4 < T_1 < T_2 < T_3$
- ☐ $T_2 < T_1 < T_4 < T_3$
- ☐ $T_4 < T_3 < T_2 < T_1$
- (b) El tiempo de ejecución para el algoritmo de clasificación por montículos (“quicksort”) es $O(n \log(n))$ (n es el número de elementos a ordenar) ...
- ☐ ... siempre.
- ☐ ... cuando el vector ya está ordenado.
- ☐ ... nunca.
- ☐ ... en el caso promedio.
- (c) Una ventaja del método de clasificación por selección, en comparación con otros algoritmos lentos, es que realiza sólo n intercambios...
- ☐ ... a veces.
- ☐ ... cuando el vector está ordenado.
- ☐ ... siempre.
- ☐ ... cuando el vector está desordenado.
- (d) ¿Cuál es el tiempo de ejecución del procedimiento de clasificación por *incrementos decrecientes* (*shell-sort*) en el caso promedio?
- ☐ $O(n^{1.3})$
- ☐ $O(n^{1.5})$
- ☐ $O(\log n)$
- ☐ $O(n!)$