Apellido y Nombre: _____

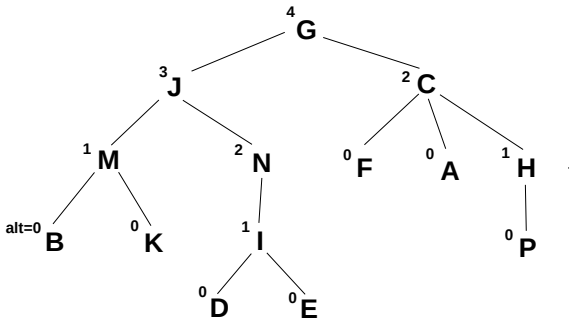
Carrera: _____ DNI: _____
[Llenar con letra mayúscula de imprenta GRANDE]

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. [26 de Febrero de 2004]

Ej. 1.- Escribir las funciones primitivas del `TAD CONJUNTO` implementado mediante *listas enlazadas clasificadas*. Es decir, implementar en Pascal los siguientes procedimientos/funciones listados abajo. Incluir todas las definiciones de tipo necesarias. `ANULA(A)`, `UNION(A,B,C)`, `INTERSECCION(A,B,C)`, `DIFERENCIA(A,B,C)`, `MIEMBRO(x,A)`, `MIN(A)`, `INSERTA(x,A)` y `SUPRIME(x,A)`.

Ej. 2.- Recordemos que la *altura* de un nodo en un árbol, es la máxima longitud de los caminos que van desde el nodo a una hoja descendiente del mismo. Por ejemplo, consideremos el árbol de la figura. El número que está arriba y a la izquierda de cada nodo es la altura del nodo.



Escribir una función `function CUENTA_ALT(n:nodo; m:integer; var altura:integer; A:arbol) : integer;` que dado un nodo `n` en un árbol `A` cuenta el número de nodos del subárbol de `A` cuya raiz es `n` tales que su altura es menor o igual que `m`. Además, en la variable `altura` debe retornar la altura del nodo `n`. Por lo tanto las siguientes llamadas deben retornar, para el árbol del ejemplo,

```
CUENTA_ALT(G,0,altura,G) -> 7
CUENTA_ALT(G,1,altura,G) -> 10
CUENTA_ALT(G,2,altura,G) -> 12
CUENTA_ALT(G,3,altura,G) -> 13
CUENTA_ALT(G,4,altura,G) -> 14
```

En todos estos casos, la variable `altura` debe retornar `altura=4`. Por otra parte, si nos referimos al nodo `N`, entonces debe retornar

```
CUENTA_ALT(N,0,altura,G) -> 2
CUENTA_ALT(N,1,altura,G) -> 3
CUENTA_ALT(N,2,altura,G) -> 4
```

y `altura=2`.

Usar las primitivas de árbol ordenado orientado siguientes: `HIJO_MAS_IZQ(n,A)`,`HERMANO_DER(n,A)`. Hacer la función *recursiva*. Notar que el conteo de nodos que cumple la condición para un nodo dado es igual a la suma sobre los hijos más 1 o 0, dependiendo de si la altura del nodo es menor que `m` o no.

Ej. 3.- Ejercicios básicos sobre TAD's

a) Escribir un procedimiento `procedure STRIDE(var L1:lista; L:lista; c, f, i: integer);` que retorna en `L1` todos le elementos en las posiciones desde el comienzo `c` hasta el fin `f` (pero sin incluirlo) a intervalos `i`. Es decir, debe retornar los elementos en las posiciones `c, c+i, c+2*i, ...`, mientras éstas sean menores que `f` y esten dentro del rango de posiciones válidas en la lista. Por ejemplo, si `L=(3,2,1,5,4,2,3,2,6)` y llamamos `STRIDE(L1,L,2,9,3)`, entonces debe retornar `L1=(2,4,2)`, mientras que si hacemos `STRIDE(L1,L,3,20,4)` debe retornar `L1=(1,3)` 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)`.

b) Escribir una función `function SUMA_COLA(C:cola): integer;` que calcula la suma de los elementos de una cola de enteros. La cola original debe quedar inalterada. *Sugerencia:* usar una variable auxiliar (cola o lista). Utilizar las primitivas del **TAD COLA:** `ANULA(C)`, `PONE_EN_COLA(x,C)`, `QUITA_DE_COLA(C)`, `VACIA(C)`, y `FRENTE_DE_COLA(C)`.

Ej. 4.- **[LIBRES] Ejercicios operativos:**

a) **Árboles:** Dibujar el árbol ordenado orientado cuyos nodos, listados en orden previo y posterior son

- `ORD_PRE` $=\{M, G, Z, R, N, A, B, D, E\}$.
- `ORD_POST` $=\{Z, R, G, A, D, E, B, N, M\}$.

b) **[LIBRES]** Dados los caracteres siguientes con sus correspondientes probabilidades, contruir el código binario y encodar la palabra `PACIENCIA` $P(P) = 0{,}3, P(A) = 0{,}1, P(C) = 0{,}3, P(I) = 0{,}05, P(E) = 0{,}05, P(N) = 0{,}2$ Calcular la longitud promedio del código obtenido.

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 $T_1(n) = 2\sqrt{n} + 0{,}5n!$, $T_2(n) = n^2 + 5n^3$, $T_3(n) = \log n + 2n$ y $T_4(n) = n^2 + 3\sqrt{n}$ decir cuál de los siguientes ordenamientos es el correcto

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

b) Sea una lista `L=(1,3,5,4)` simplemente enlazada por punteros o cursores, y sea `p` la posición correspondiente al elemento 3. Despues de hacer `SUPRIME(p,L)`, ¿cual es el resultado de hacer `x = RECUPERA(p,L)`? ...

- ☐ ... retorna el elemento 1
- ☐ ... produce un error
- ☐ ... retorna el elemento 5
- ☐ ... retorna el elemento 3

c) ¿Cuál es el criterio para elegir una buena función de dispersión? ...

- ☐ Debe tratar de concentrar los elementos en pocas cubetas.
- ☐ Debe tratar de concentrar los elementos en una sóla cubeta.
- ☐ Debe tratar de concentrar los elementos en la primera cubeta.
- ☐ Debe distribuir los elementos en la forma más uniforme posible entre las cubetas.

d) ¿Cuál de los siguientes algoritmos de clasificación es el más rápido *en el caso promedio*?

- ☐ Burbuja (*"Bubble-sort"*)
- ☐ Clasificación por *incrementos decrecientes* (*shell-sort*)
- ☐ Selección
- ☐ Clasificación rápida (*"Quick-sort"*)