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.
2do Parcial. [26 de mayo de 2005]

[Ej. 1] **[clases (20 puntos)]** Escribir la implementación en `C++` del `TAD ARBOL ORDENADO ORIENTADO` (clase `tree`). Las funciones a implementar son `insert(p,x)`, `erase(p)`, `find(x)` y `clear()`. Observaciones:

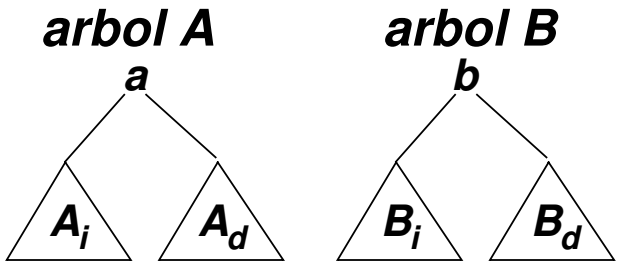
- En caso de optar por escribir la interfase *"básica"*, debe escribir todas las declaraciones necesarias de la clase, tanto en la parte privada como pública.
- En caso de optar por la interfase *"avanzada"*, debe declarar e implementar completamente las partes privadas de la clase `tree`, `iterator` y `cell`.

[Ej. 2] **[programacion (50 pts)]**

a) **[es-menor (30 pts)]** Se define una relación de orden entre árboles binarios de enteros de la siguiente forma:

$$A < B \text{ si } \begin{cases} a < b; \\ a = b \ \text{ y } \ A_i < B_i \\ a = b \ \text{ y } \ A_i = B_i \ \text{ y } \ A_d < B_d \end{cases} \qquad \text{(si } a \text{ o } b \text{ son } \Lambda \text{ entonces se considera } -\infty)$$

donde a, b son las raíces y A_i, A_d, B_i, B_d son los subárboles izquierdos y derechos.
Consigna: Escribir una función `bool es_menor(tree<int>&A,tree<int>&B)` que retorna verdadero si $A < B$.



b) **[count-if (10 pts)]** Escribir una función `int count_if(tree<int>&T,bool (*pred)(int x));` que retorna el número de nodos del árbol `T` que satisfacen el predicado `pred`. Por ejemplo, si `T=(1 2 (3 5 7 6) 4)`, entonces `count_if(T,odd)` debe retornar 4. Escribir el predicado `bool odd(int x)` que determina si un entero es impar.

c) **[list-if (10 pts)]** Escribir una función `void list_if(tree<int>&T,list<int>&L,bool (*pred)(int x));` que retorna en `L` la lista de valores nodales en orden previo de un árbol ordenado orientado `T` que satisfacen el predicado `pred`. Por ejemplo, si `T=(1 (-2 7 (8 -7) (3 -5 -6)))`, entonces después de `list_if(T,L,positive)`, debe quedar `L={1,7,8,3}`. Escribir el predicado `bool positive(int x)` que determina si un entero es mayor que 0.

[Ej. 3] **[operativos (20 pts)]**

- **[rec-arbol (8 pts)]** Dibujar el árbol ordenado orientado cuyos nodos, listados en orden previo y posterior son
  - `ORD_PRE` $=\{Z, A, R, Q, L, M, N, T, S, W, Q\}$,
  - `ORD_POST` $=\{Q, L, R, N, M, A, W, Q, S, T, Z\}$.

- **[huffman (8 pts)]** Dados los caracteres siguientes con sus correspondientes probabilidades, contruir el código binario y encodar la palabra `CONCLAVE`
  $P(C) = 0.05, P(O) = 0.05, P(N) = 0.1, P(A) = 0.2, P(L) = 0.2, P(V) = 0.1, P(E) = 0.1, P(Q) = 0.2$
  Calcular la longitud promedio del código obtenido.
- **[part-arbol (4 pts)]** Particione el árbol AOO `(p (q (s x y) t u) (r v w))` con respecto al nodo `q`, es decir indique cuales son sus antecesores y descendientes propios, derecha e izquierda.

[Ej. 4] **[preguntas (10 pts, 2.5 por pregunta)]** 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) ¿Como es el número de niveles l para un árbol binario lleno, en función del número de nodos n?

- ☐ ... $l = O(2^n)$
- ☐ ... $l = O(n!)$
- ☐ ... $l = O(\log_2 n)$
- ☐ ... $l = O(n^2)$

---

b) En la codificación de mensajes en secuencias de bits, la condición de prefijos asegura que...

- ☐ ... el código es óptimo.
- ☐ ... el mensaje codificado se puede *decodificar* en forma única.
- ☐ ... el mensaje se puede *codificar* en forma única.
- ☐ ... el código no es redundante.

---

c) El algoritmo de Huffmann para compresión se basa en ...

- ☐ ... utilizar códigos más cortos para los caracteres con *mayor* probabilidad.
- ☐ ... utilizar códigos más cortos para los caracteres con *menor* probabilidad.
- ☐ ... utilizar códigos que generen longitudes promedio lo más largas posibles.
- ☐ ... utilizar códigos con la misma longitud en bits para cada carácter.

---

d) Considere el AOO `Q=(3 5 8 (7 8 6) 9)`, después de hacer las siguientes operaciones

```
tree<int>::iterator n = Q.find(5);
n++;
n = Q.erase(n);
n++;
n = Q.erase(n);
```

- ☐ ... queda `(3 5 (7 8 6))`, `*n=7`
- ☐ ... queda `(3 5 9)`, `n=end()`
- ☐ ... da un error.
- ☐ ... queda `(3 5 (7 8 6))`, `n=end()`
