Algoritmos y Estructuras de Datos.
Examen Final. [06 de Julio de 2006]

[Ej. 1] **[Clases (20 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] **[programacion (total = 80 puntos)]**

a) **[encuentra (35 puntos)]** Escribir una función
`bool encuentra(list<int> &L1, list<int> &L2, list<int> &indx);` que,

- Retorna `true` o `false` dependiendo de si `L1` es una sublista o no de `L2`.
- En caso de que si lo sea, retorna en `indx` los índices de los elementos de `L2` que forman `L1`, si no `indx` debe retornar vacía, independientemente de lo que contenía previamente.

Por ejemplo, si `L2`=$\{13, 9, 8, 12, 9, 6, 12, 2, 9, 14, 18, 10\}$ y `L1`=$\{13, 9, 9, 6, 2, 14\}$ entonces `encuentra` debe retornar `true`, e `indx`=$\{0, 1, 4, 5, 7, 9\}$. Si `L1`=$\{8, 9, 13\}$ debe retornar `false` e `indx`=$\{\}$. Nota: Los índices en `indx` deben ser estrictamente crecientes. *Restricciones:* No usar estructuras auxiliares. El tiempo de ejecución del algoritmo debe ser $O(n)$.

b) **[encuentra-suma (35 puntos)]** Escribir una función
`tree<double>::iterator encuentra_suma(double s, tree<double> &A);` que retorna el nodo `m` del árbol orientado `A` tal que la suma de las etiquetas de todos los nodos descendientes de `m` (incluyendo a `m`) es `s`. Si no existe un tal nodo, entonces debe retornar Λ. Si hay varios nodos que cumplen la condición basta con que retorne uno de ellos. Por ejemplo, para `A=(10 21 (6 (8 7) (11 (12 3) 6) (4 6 5)))`

- `encuentra_suma(32,A)` retorna el nodo 11,
- `encuentra_suma(15,A)` puede retornar el nodo 8, el 4 o el 12.
- `encuentra_suma(9,A)` retorna Λ

Restricciones: El algoritmo debe ser $O(n)$ donde n es el número de nodos en el árbol.

c) **[reordena-pila (10 puntos)]** Escribir una función
`void reordena(stack<int> &P,bool (*pred)(int));` que reordena los elementos de una pila de tal forma que quedan los que no satisfacen el predicado `pred()` en el fondo y los que si lo satisfacen arriba. Por ejemplo si $P = \{1, 3, 4, 2, 3, 5, 7, 6, 8, 2, 9\}$ y `pred()` es `bool par(int x) { return x%2==0; }` entonces debe quedar $P = \{4, 2, 6, 8, 2, 1, 3, 3, 5, 7, 9\}$. *Restricciones:* el algoritmo debe ser *estable*, es decir los elementos que satisfacen `pred()` deben quedar en el mismo orden entre sí, y lo mismo para los que no satisfacen `pred()`. Se pueden usar dos estructuras auxiliares (listas, pilas o colas).

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

a) **[rec-arbol (40 ptos)]** Dibujar el árbol ordenado orientado cuyos nodos, listados en orden previo y posterior son

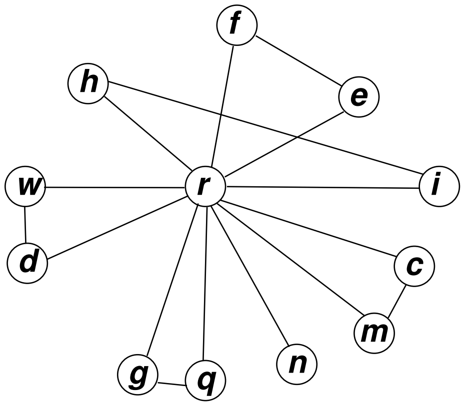
- `ORD_PRE` $=\{A, B, C, E, F, Z, Y, W, G, D\}$,
- `ORD_POST` $=\{B, E, Z, Y, W, F, G, C, D, A\}$.

b) **[misc-arbol (20pt)]:** Dado el árbol `(b t (u (z x y) w))`,

1) Cuál es el nodo de profundidad 2 que está a la izquierda de `w`?
2) Particione el árbol con respecto al nodo `u`, es decir indique cuales son sus antecesores y descendientes propios, derecha e izquierda.

c) **[colorear-grafo (20 ptos)]**

Colorear el siguiente grafo, utilizando una estrategia heurística para tratar de usar el menor número de colores posibles.



[Ej. 4] **[Preguntas (total = 20 puntos, 5puntos 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**!!]

Sea el árbol `(q e (r w) s t)`. ¿Cuál de las opciones es verdadera?

- [ ] Tiene 4 nodos a profundidad 1 y 5 hojas.
- [ ] Tiene dos raíces y 4 hojas.
- [ ] Tiene 4 nodos a profundidad 1 y 4 hojas.
- [ ] Tiene 5 nodos a profundidad 1 y 4 hojas.

¿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 $O(n \log n)$.

Dado el árbol binario `(z (a b q) r)`, ¿cuál de las siguientes opciones es verdadera?

- [ ] Es completo y es lleno.
- [ ] Es completo pero no lleno.
- [ ] Es lleno pero no completo.
- [ ] Ni es completo ni es lleno.

Sea un tabla de dispersión abierta con B cubetas y n elementos. Asumiendo que la función de dispersión es lo suficientemente buena como para distribuir los elementos en forma uniforme entre las cubetas, el costo medio de inserción de un nuevo elemento es

- [ ] $O(n^2/B)$
- [ ] $O((n/B)^2)$
- [ ] $O(n+B)$
- [ ] $O(n/B)$