Instituto de Computación. Facultad de Ingeniería. Universidad de la República

Primer Parcial de Programación 2

Mayo de 2021

Problema 1

Considere la siguiente definición del tipo ***ABB*** de árboles binarios de búsqueda de enteros (sin elementos repetidos) y la definición del tipo ***Lista*** para listas de enteros:

```
struct nodoABB{
   int dato;
   nodoABB *izq, *der;
}
typedef nodoABB * ABB;
```

```
struct nodoLista{
   int dato;
   nodoLista *sig;
}
typedef nodoLista * Lista;
```

Implemente un procedimiento recursivo ***rangoEnLista*** que dados un árbol binario de búsqueda *t* de tipo *ABB*, dos enteros *inf* y *sup*, y una lista *l* de tipo *Lista*, agregue en *l* (que asumimos inicialmente vacía: NULL) los elementos de *t* que sean menores estrictos que *sup* y mayores estrictos que *inf* (asumimos *inf* < *sup*). Los elementos incorporados en *l* deben estar ordenados de menor a mayor. El procedimiento debe tener O(*n*) peor caso, siendo *n* la cantidad de nodos de *t*, aunque deben evitarse recorrer nodos de *t* que resulten innecesarios. Si no hay elementos de *t* en el rango entre *inf* y *sup* (en particular si *t* es vacío: NULL), el procedimiento no tendrá efecto. No defina operaciones auxiliares para implementar *rangoEnLista*.

```
void rangoEnLista (ABB t, int inf, int sup, Lista & l)
```

Problema 2

Considere un árbol general de enteros representado mediante un árbol binario de enteros con la semántica: puntero al primer hijo (pH), puntero al siguiente hermano (sH).

```
struct nodoAG{
    int dato;
    nodoAG *pH, *sH;
}
typedef struct nodoAG * AG;
```

Un árbol general *t* es *n ario* (con *n*>0) si y solo si *t* es vacío (NULL) ó cada elemento (nodo) de *t* tiene a lo sumo *n* hijos (entre 0 y *n* nodos).

Implemente la función ***bool nArio (AG t, int n)*** que retorne true si y solo si el árbol *t* es *n ario*, asumiendo que *n*>0 y t->sH==NULL. Si utiliza operaciones auxiliares, deberá implementarlas.

Instituto de Computación. Facultad de Ingeniería. Universidad de la República

Primer Parcial de Programación 2

Mayo de 2021

Problema 1

Considere la siguiente definición del tipo ***ABB*** de árboles binarios de búsqueda de enteros (sin elementos repetidos) y la definición del tipo ***Lista*** para listas de enteros:

```
struct nodoABB{
   int dato;
   nodoABB *izq, *der;
}
typedef nodoABB * ABB;
```

```
struct nodoLista{
   int dato;
   nodoLista *sig;
}
typedef nodoLista * Lista;
```

Implemente un procedimiento recursivo ***rangoEnLista*** que dados un árbol binario de búsqueda *t* de tipo *ABB*, dos enteros *inf* y *sup*, y una lista *l* de tipo *Lista*, agregue en *l* (que asumimos inicialmente vacía: NULL) los elementos de *t* que sean menores estrictos que *sup* y mayores estrictos que *inf* (asumimos *inf* < *sup*). Los elementos incorporados en *l* deben estar ordenados de menor a mayor. El procedimiento debe tener O(*n*) peor caso, siendo *n* la cantidad de nodos de *t*, aunque deben evitarse recorrer nodos de *t* que resulten innecesarios. Si no hay elementos de *t* en el rango entre *inf* y *sup* (en particular si *t* es vacío: NULL), el procedimiento no tendrá efecto. No defina operaciones auxiliares para implementar *rangoEnLista*.

```
void rangoEnLista (ABB t, int inf, int sup, Lista & l)
```

Problema 2

Considere un árbol general de enteros representado mediante un árbol binario de enteros con la semántica: puntero al primer hijo (pH), puntero al siguiente hermano (sH).

```
struct nodoAG{
    int dato;
    nodoAG *pH, *sH;
}
typedef struct nodoAG * AG;
```

Un árbol general *t* es *n ario* (con *n*>0) si y solo si *t* es vacío (NULL) ó cada elemento (nodo) de *t* tiene a lo sumo *n* hijos (entre 0 y *n* nodos).

Implemente la función ***bool nArio (AG t, int n)*** que retorne true si y solo si el árbol *t* es *n ario*, asumiendo que *n*>0 y t->sH==NULL (si el árbol *t* no es vacío, el nodo raíz no tiene hermanos). Si utiliza operaciones auxiliares, deberá implementarlas.

Instituto de Computación. Facultad de Ingeniería. Universidad de la República

Primer Parcial de Programación 2

Mayo de 2021

Soluciones

Problema 1

```
void rangoEnLista (ABB t, int inf, int sup, Lista & l){
      /* Recorrido inorder invertido (subárbol derecho antes que subárbol
      izquierdo) con inserciones al comienzo de l, para generar en O(n) una
      lista ordenada crecientemente. */
      if (t != NULL){
            if (t->dato < sup-1)
                  rangoEnLista (t->der, inf, sup, l);
            if (t->dato > inf && t->dato < sup){
                  Lista nodoX = new nodoLista;
                  nodoX->dato = x;
                  nodoX->sig = l;
                  l = nodoX;
            }
            if (t->dato > inf+1)
                  rangoEnLista (t->izq, inf, sup, l);
      }
}
```

Problema 2

```
bool nArio (AG t, int n){
      if (t == NULL)
            return true;
      else
            return (hijos(t) <= n) && nArio(t->pH, n) && nArio(t->sH, n);
}

// Retorna la cantidad de hijos de t en un árbol pH-sH. Precondición: t != NULL.
int hijos (AG t){
      int cantHijos = 0;
      t = t->pH;
      while (t != NULL){
            cantHijos++;
            t = t->sH;
      }
      return cantHijos;
}
```