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

Primer Parcial de Programación 2

Mayo de 2022

Problema 1 (11 puntos)

Considere la siguiente definición del tipo ***Lista*** para listas de enteros:

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

Implemente una función iterativa ***sinRepetidos*** que dada una lista *l* de tipo *Lista* que puede contener valores exclusivamente en el rango [0 : p] (entre 0 y p inclusive, con p > 0), retorne *true* si y solo si la lista no tiene elementos repetidos. Si la lista es vacía (NULL), el resultado debe ser *true*. Se pueden usar estructuras de datos auxiliares, manejando adecuadamente la memoria (pedido y liberación, si corresponde). La función debe ser O(*p*) en el peor caso, aunque no es necesario justificar el cumplimento del orden exigido.

```
PRE: Cada elemento x de la lista l cumple: 0 ≤ x ≤ p, con p > 0
bool sinRepetidos(Lista l, int p)
```

Problema 2 (12 puntos)

Considere la siguiente definición para árboles binarios de enteros (*AB*):

```
struct nodoAB{
   int dato;
   nodoAB *izq, *der;
}
typedef struct nodoAB * AB;
```

Implemente una función recursiva (sin usar iteración) ***maximo*** que dado un árbol binario *t* de tipo *AB* retorne un puntero al nodo de *t* que tenga el dato mayor (el máximo), o NULL si *t* es NULL. Asumimos que *t* no tiene elementos repetidos, aunque no puede asumirse que los elementos en *t* están ordenados como en un árbol binario de búsqueda.

No defina ni asuma la existencia de operaciones auxiliares para implementar *maximo*. Además, la función *maximo* no deberá visitar cada nodo de *t* más de una vez (solo se puede recorrer el árbol una vez).

```
PRE: t no tiene elementos repetidos
AB maximo(AB t)
```

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

Primer Parcial de Programación 2

Mayo de 2022

Problema 3 (12 puntos)

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;
```

Implemente un procedimiento recursivo (sin usar iteración) ***nivelEnLista*** que dados un árbol general *t* de tipo *AG*, un entero *k* (*k*>0) y una lista *l* de tipo *Lista* (del *Problema 1*), agregue en *l* (que asumimos inicialmente vacía: NULL) los elementos de *t* que estén en el nivel *k* de *t*. Si no hay elementos de *t* en el nivel *k*, en particular si *t* es vacío (NULL), el procedimiento no tendrá efecto (la lista quedará vacía: NULL). Los elementos incorporados en *l* pueden estar en cualquier orden; esto es, solo se pide que sean elementos del nivel *k* de *t*. Recuerde que t->sH==NULL (la raíz del árbol no tiene hermanos) y que en un árbol no vacío la raíz se encuentra en el nivel 1 (uno).

El procedimiento puede visitar cada nodo de *t* a lo sumo una vez, aunque debe evitarse visitar nodos que resulten innecesarios. No defina operaciones auxiliares para implementar *nivelEnLista*.

```
PRE: k>0
void nivelEnLista(AG t, Lista & l, int k)
```

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

Primer Parcial de Programación 2

Mayo de 2022

Soluciones

Problema 1

```
PRE: los elementos de l están en el rango [0:p] y p>0
bool sinRepetidos(Lista l, int p){
      bool*  Pertenece = new bool[p+1];
      for (int i=0; i<p+1; i++)
            Pertenece[i] = false;
      while (l!=NULL && !Pertenece[l->dato]){
            Pertenece[l->dato] = true;
            l = l->sig;
      }
      delete [] Pertenece;
      return (l==NULL);
}
```

Problema 2

```
PRE: el AB t no tiene elementos repetidos
AB maximo(AB t){
      if (t==NULL) return NULL;
      else{
            AB nodoMax = t; // inicializamos con la raíz
            AB nodoMaxRec = maximo(t->izq);
            if (nodoMaxRec!=NULL && nodoMaxRec->dato > nodoMax->dato)
                  NodoMax = nodoMaxRec; // se considera el max de t->izq
            nodoMaxRec = maximo(t->der);
            if (nodoMaxRec!=NULL && nodoMaxRec->dato > nodoMax->dato)
                  NodoMax = nodoMaxRec; // se considera el max de t->der
            return nodoMax;
      }
}
```

Problema 3

```
PRE: k>0
void nivelEnLista(AG t, Lista & l, int k)
      if (t!=NULL && k>0){
            if (k==1){ // inserta al comienzo de l el elemento t->dato
                  Lista nuevo = new nodoLista;
                  nuevo->dato = t->dato;
                  nuevo->sig = l;
                  l = nuevo;
            }
            else nivelEnLista(t->pH, l, k-1);
            nivelEnLista(t->sH, l, k); // Se hace tanto si k==1 como si k>1
      }
}
```