CERTAMEN #2 ICI-142

Wenceslao Palma, Laura Griffiths

1. (30 ptos.) Considerando la siguiente definición de nodo:

```
typedef struct node{
  struct node *sgte;
  int v;
}nodo;
```

Escriba la función **int busquedaBinaria(nodo *,int)**, la cual determina, mediante búsqueda binaria, si un número existe en una lista enlazada apuntada por ***INICIO**. Restricción: la solución debe considerar únicamente el uso de punteros. No está permitido el uso de arreglos ni el uso de variables de tipo entero.

```
(a) Uso de funciones (5 ptos)
(b) Uso de punteros (5 ptos)
(c) Solución (20 ptos)

int busquedaBinaria(nodo *INICIO, int v){
  nodo *izq,*der,*mitad;

  izq=der=INICIO;
  while (der->sgte!=NULL)
        der = der->sgte;

  if (v>der->v) return 0;
  if (v<izq->v) return 0;

  while (der->sgte!=izq){
        mitad = centro(izq,der);
        if (v>mitad->v)
            izq = mitad->sgte;
        else
            if (v< mitad->v){
               antMitad = izq;
               while (antMitad->sgte!=mitad)
                    antMitad = antMitad->sgte;
               der=antMitad;
            }else
               return 1;
  }
  return 0;
}
```

```
nodo * centro(nodo *izq,nodo *der){

      struct nodo *ant,

      while (ant->sgte!=der)
            ant = ant->sgte;

      while (izq!=der) && (der->sgte=!izq){

         izq = izq->sgte;
         der = ant;
         ant = izq;

         while (ant->sgte!=der) && (ant->sgte!=NULL)
               ant = ant->sgte;

     }
     return (izq);
}
```

2. (30 ptos.) En un archivo llamado "a.txt" se encuentra almacenado un conjunto de números enteros sin elementos repetidos. De la misma forma en otro archivo llamado "b.txt" se encuentra almacenado un conjunto de números enteros sin elementos repetidos. Escriba las siguientes funciones:

   - **void crearListaA(nodo **);** la cual crea una lista enlazada, apuntada por ***A**, a partir de los números almacenados en "a.txt". El contenido de la lista se encuentra en orden ascendente. Para el ordenamiento no se permite el uso de estructuras de datos auxiliares, se debe utilizar sólo la lista enlazada.
   - **void crearListaB(nodo **);** la cual crea una lista enlazada, apuntada por ***B**, a partir de los números almacenados en "b.txt". El contenido de la lista se encuentra en orden descendente. Para el ordenamiento no se permite el uso de estructuras de datos auxiliares, se debe utilizar sólo la lista enlazada.
   - **node * crearListaC(nodo **,nodo **);** la cual genera la mezcla de ambas listas (apuntada por ***C**) en orden ascendente sin crear nuevos nodos.

Función **void crearListaA(nodo **);** (8 ptos)

```
void crearListaA(nodo **A){
  FILE *fp;
  nodo *nuevo;

  fp = fopen("a.txt","r");
  if (fp==NULL){printf("error en la apertura del archivo .....\n");exit(0);}

  while (!feof(fp)){
     nuevo = (nodo *)malloc(sizeof(nodo));
     fscanf(fp,"%d \n",&nuevo->v);
     nuevo->sgte=NULL;

     if (*A==NULL)
         *A=nuevo;
     else{
         tmp=*A;
         if (nuevo->v<=tmp->v){
             nuevo->sgte=tmp;
             *A = nuevo;
         }else{
             insert = 0;
             while ((tmp->sgte!=NULL) && (!insert)){
                   if ((nuevo->v >= tmp->v) && (nuevo->v<=(tmp->sgte)->v)){
                       nuevo->sgte = tmp->sgte;
                       tmp->sgte = nuevo;
                       insert=1;
                   }else
                       tmp=tmp->sgte;
             }
             if (!insert)
                tmp->sgte = nuevo;
         }
     }
  }
}
```

Función **void crearListaB(nodo **);** (8 ptos)

```
void crearListaA(nodo **B){
  FILE *fp;
  nodo *nuevo;

  fp = fopen("b.txt","r");
  if (fp==NULL){printf("error en la apertura del archivo .....\n");exit(0);}

  while (!feof(fp)){
     nuevo = (nodo *)malloc(sizeof(nodo));
     fscanf(fp,"%d \n",&nuevo->v);
     nuevo->sgte=NULL;

     if (*B==NULL)
         *B=nuevo;
     else{
         tmp=*B;
         if (nuevo->v>=tmp->v){
             nuevo->sgte=tmp;
             *B = nuevo;
         }else{
               insert = 0;
               while ((tmp->sgte!=NULL) &&  (!insert)){
                  if ((nuevo->v <= tmp->v) && (nuevo->v>=(tmp->sgte)->v)){
                     nuevo->sgte = tmp->sgte;
                     tmp->sgte = nuevo;
                     insert=1;
                  }else
                     tmp=tmp->sgte;
               }
               if (!insert)
                   tmp->sgte = nuevo;
         }
     }
  }
}
```

Función **node * crearListaC(nodo **,nodo **);** (14 ptos)

```
invertir(&B);
node * crearListaC(nodo **A,nodo **B){
  nodo *C;
     struct nodo *c,t,t2,t3;

     t = a;
     t2 = b;

     if (a->dato < b->dato){
           c = a;
           t = t->sgte;
     }else{
           c = b;
   t2 = t2->sgte;
     }
     t3 = c;

     while (t!=NULL) && (t2!=NULL) {
         if (t->dato < t2->dato){
            t3->sgte = t;
            t3 = t;
            t = t->sgte;
         }else{
            t3->sgte = t2;
            t3 = t2;
            t2 = t2->sgte;
         }
     }
     if (t==NULL){
        while (t2!=NULL){
            t3->sgte = t2;
            t3 = t2;
            t2 = t2->sgte;
        }
     }

     if (t2==NULL){
        while (t!=NULL){
           t3->sgte = t;
           t3 = t;
           t = t->sgte;
        }
     }
     t3->sgte = NULL;

     return c;
}
```

```
void invertir(nodo **INICIO){
  nodo *actual,*antecesor,*sucesor;

  antecesor=NULL;
  actual=*INICIO;

  while (actual!=NULL){
       sucesor=actual->sgte;
       actual->sgte=antecesor;
       antecesor=actual;
       actual=sucesor;
  }
  *INICIO=antecesor;
}
```