UNIVERSIDAD DE MÁLAGA
Dpto. Lenguajes y CC. Computación
ETSI Informática

Laboratorio de Programación III

Examen Diciembre 2000

Alumno:
Grupo: Especialidad: Código Ordenador:

1. Construir el TAD lista posicional con una implementación ***acotada con cota variable***, según el siguiente módulo de definición:

```
DEFINITION MODULE ListaPos;
FROM TadItem IMPORT ITEM;

TYPE  LISTAPOS;
      TIPO_ERROR = <<a definir>>;

      PROCEDURE Error():  TIPO_ERROR;

      PROCEDURE Crear(cota: CARDINAL): LISTAPOS;

      PROCEDURE Longitud(l: LISTAPOS): CARDINAL;

      PROCEDURE ReAcotar(VAR l: LISTAPOS; nueva_cota: CARDINAL);

      (* 1 <= i <= Longitud(l) + 1 *)
      PROCEDURE Insertar(VAR l: LISTAPOS;i: CARDINAL;x: ITEM);

      (* 1 <= i <= Longitud(l) *)
      PROCEDURE Cambiar(VAR l: LISTAPOS;x: ITEM;i: CARDINAL);

      (* 1 <= i <= Longitud(l) *)
      PROCEDURE Eliminar(VAR l: LISTAPOS;i: CARDINAL);

      (* (1 <= i <= Longitud(l)) *)
      PROCEDURE Elemento(l: LISTAPOS;i: CARDINAL): ITEM;

      PROCEDURE Destruir(VAR l: LISTAPOS);

END ListaPos.
```

Puesto que la implementación es ***acotada***, sólo puede haber un número máximo de elementos contenidos en la lista posicional. Este número máximo de elementos se establece al crear la lista posicional a través del parámetro **`cota`** del procedimiento **`Crear`**. Sin embargo, la cota es ***variable***, lo que significa que se puede alterar en tiempo de ejecución. De esto se encarga el procedimiento **`ReAcotar`**, que establece **`nueva_cota`** como nuevo número máximo de elementos de la lista posicional. Si la nueva cota es menor que la longitud actual de la lista posicional, entonces ésta debe truncarse.

Las listas posicionales se representarán mediante siguiente estructura de datos:

```
TYPE  PNODO = POINTER TO Nodo;
      Nodo =RECORD
               cont : ITEM;
               sig : PNODO;
            END;
      LISTAPOS = POINTER TO Cabecera;
      Cabecera = RECORD
                   cota: CARDINAL;
                   longitud: CARDINAL;
                   primero: PNODO;
                 END;
```

2. Implementar la siguiente biblioteca para listas posicionales

```
DEFINITION MODULE BibLisP;
FROM TadItem IMPORT ITEM;
FROM ListaPos IMPORT LISTAPOS;

PROCEDURE Escribir(l: LISTAPOS);
PROCEDURE Maximo(l: LISTAPOS): ITEM;
PROCEDURE Iguales(l1, l2: LISTAPOS): BOOLEAN;
PROCEDURE Concatenar(VAR a: LISTAPOS; b,c: LISTAPOS);

END BibLisP.
```

3. Escribir un programa de prueba para el TAD y la biblioteca (tomando **CHAR** como tipo **ITEM**) de acuerdo con el siguiente menú:

```
1) Crear
2) Longitud
3) Reacotar
4) Insertar
5) Cambiar
6) Eliminar
7) Elemento
8) Escribir
9) Máximo
10) Iguales
11) Concatenar
12) Destruir
13) Salir
```