```
#ifndef NODO_H
#define NODO_H

#ifndef  NO_ECONTRADO
#define NO_ENCONTRADO -1
#endif // ! NO_ECONTRADO

#ifndef LINEA
#define LINEA "-------------------------------------------------------------------------------------------
--------------------"
#endif // !LINEA

typedef void* ELEMENTO;

struct Nodo {
   ELEMENTO dato;
   Nodo* siguiente;
};

//-----------------------------------------------------Funciones-----------------------------------------
--------
/*
   PRE: La lista debe haber sido creada.
   POST: Agrego al inicio de la lista 1 elemento.
 */
void insertarAlInicio(Nodo*&, ELEMENTO);

/*
   PRE: La lista debe haber sido creada.
   POST: Agrego 1 elemento al final de la lista.
 */
```

```
void insertarAlFinal(Nodo*&, ELEMENTO);

/*
   PRE: La lista debe haber sido creada.
   POST: Agrego 1 elemento a la lista en la posicion indicada si es que existe.
 */
void insertarElementoEnPosicion(Nodo* &, int iPosicion, ELEMENTO dato);

/*
   PRE: La lista debe haber sido creada.
   POST: Busca un elemento en la lista y me indica en que posicion se encuetra.
 */
int buscarElemento(Nodo*, ELEMENTO);

/*
   PRE: La lista debe haber sido creada.
   POST: Almaceno en una variable el ELEMENTO que poseo en una posicion indicada de la lista si es que existe.
 */
void obtenerElemento(Nodo*, int iPosicion, ELEMENTO&);

/*
   PRE: La lista debe haber sido creada.
   POST: Remuevo y almaceno en una variable el ELEMENTO que poseo en el inicio de la lista.
 */
void quitarElementoDelInicio(Nodo*&, ELEMENTO&);

/*
   PRE: La lista debe haber sido creada.
```

POST: Remuevo y almaceno en una variable el ELEMENTO que poseo en una posicion indicada de la lista si es que existe.

***/**

void quitarElementoDePosicion(Nodo*&, int iPosicion, ELEMENTO &);

/*

PRE: La lista debe haber sido creada.

POST: La lista queda vacia.

***/**

void vaciarLista(Nodo*&);

/*

PRE: La lista debe haber sido creada.

POST: Indica si la lista tiene o no elementos que la conpongan.

***/**

bool listaVacia(Nodo*);

#endif // !NODO_H