

Práctico 6 – Estructuras: Lista

OBJETIVOS:

- Comprender el concepto de 'punteros' y utilizarlos adecuadamente.
- Trabajar con las diferentes variantes de la estructura lista.

Ejercicio 1

Implemente las siguientes funciones para la estructura de lista simplemente encadenada.

- boolean pertenece (int x);
Pos: Retorna true si el dato pasado como parámetro pertenece a la lista.
- void borrar (int x);
Pre: El elemento x pasado como parámetro pertenece a la lista.
Pos: Elimina de la lista la primer ocurrencia del elemento x.
- int largo ();
Pos: Retorna la cantidad de elementos de la lista.
- void snoc (int x);
Pos: Inserta el dato pasado como parámetro al final de la lista.

Las funciones solicitadas pueden ser implementadas en forma iterativa y recursiva, implemente ambas variantes.

Ejercicio 2

Implemente las siguientes funciones para la estructura de lista simplemente encadenada.

- Lista invertir ();
Pos: Retorna una nueva lista, resultado de invertir el orden de los elementos de la lista original.
- boolean estaOrdenada ();
Pos: Retorna true si la lista está ordenada.
- void insOrd (int elem);
Pre: La lista está ordenada ascendentemente.
Pos: Inserta el elemento pasado como parámetro de forma ordenada en la lista.
- int cuenta (int elem);
Pos: Retorna la cantidad de veces que aparece el elemento pasado como parámetro en la lista.

- e) int maximo (); Pre: La lista no es vacía.
 Pos: Retorna el máximo elemento de la lista.
- f) int promedio ();
 Pre: La lista no es vacía.
 Pos: Retorna el promedio de los elementos de la lista.
- g) int tomar_n (int n);
 Pre: El largo de la lista es mayor o igual que n.
 Pos: Retorna el elemento que se encuentra en la posición n, contando a partir de 1.
- h) Lista cambiar (int n, int m);
 Pos: Retorna la lista resultado de cambiar todas las ocurrencias de n por el elemento m.

Ejercicio 3

Implemente las siguientes funciones recursivamente.

- a) boolean iguales (Lista l, Lista p);
 Pos: Retorna true sii la lista l es igual a la lista p. Dos listas son iguales si son vacías o si tienen los mismos elementos y en el mismo orden.
- b) Lista intercalar (Lista l, Lista p);
 Pre: Las listas p y l están ordenadas.
 Pos: Retorna una nueva lista resultado de intercalar ordenadamente los elementos de p y l.
- c) Lista concatenar (Lista l, Lista p);
 Pos: Retorna una nueva lista, resultado de concatenar la list p al final de la lista l.
- d) boolean estaIncluida (Lista l, Lista p);
 Pos: Retorna true sii la lista l está incluida en la lista p. Una lista l está incluida en una lista p si es posible encontrar una sub-secuencia de elementos de p igual a la secuencia de elementos de l.

Ejercicio 4

Implemente las siguientes funciones para la estructura de lista doblemente encadenada.

- a) void Cons (Lista l, int x);
 Pos: Inserta el elemento x al principio de la lista l.
- b) void insOrd(Lista l, int x);
 Pre: La lista l está ordenada.
 Pos: Inserta el elemento x en la lista l manteniendo su orden.
- c) void borrar(int x);
 Pos: Elimina la primer ocurrencia del elemento x en la lista