Programación de Estructuras Dinámicas

Hoja de Ejercicios #2 – Listas lineales simplemente enlazadas

Elabore programas que incluyan funciones recursivas para resolver los siguientes problemas:

1) Elabore un programa que permita llenar una lista simple con *n* números aleatorios entre 1 y 100 por medio de una función invocada desde *main*. Recuerde la función rand( ) y la forma de generar una semilla para la generación de números aleatorios. Asimismo elabore funciones de impresión que trabajen de la siguiente manera:
   a) Mostrar todos los elementos de la lista.
   b) Mostrar solo los números pares de la lista.
   c) Mostrar solo los números impares de la lista.
   d) Mostrar todo su contenido en sentido inverso (esta función debe ser recursiva).

2) Dada una lista lineal simplemente enlazada, en la que en cada nodo se almacenan el coeficiente y el exponente de x de los términos de un polinomio de la forma $p(x) = a_0 x^0 + a_1 x^1 + a_2 x^2 + \ldots + a_n x^n$, elabore una función que evalúe el polinomio para un valor de *x* dado. La función miembro recibirá como argumento el valor de *x* y deberá retornará el valor de *p(x)* calculado. El tipo del valor retornado por la función debe ser *float*. Si elabora la versión recursiva de la solución, la función miembro deberá recibir también el puntero cabeza de la lista.

3) Elabore un programa que convierta un número entero en base diez, a su binario equivalente. Guarde los dígitos binarios en una lista simple y realice el despliegue de los dígitos en forma correcta.

4) Escriba un programa que muestre un menú que permita administrar una lista lineal simple con datos sobre personas. Deberá desplegar un menú con las siguientes opciones:
   a) Llenar la lista.
   b) Eliminar una persona.
   c) Actualizar los datos de una persona.
   d) Mostrar todas las persona.
   e) Salir.

   Para cada persona deberá tomar en cuenta los siguientes datos: carnet, nombre, apellido, edad, número de teléfono y correo electrónico.

5) Elabore una función recursiva que calcule la suma de los números pares y la suma de los números impares contenidos en una lista. La función debe recibe como primer argumento el puntero de la lista y actualizar otros dos argumentos con las sumas realizadas.

6) Elabore una función recursiva que reciba una lista lineal simplemente enlazada y la devuelva con los nodos colocados en forma inversa.

7) Elabore una función recursiva de tres argumentos. El primero de ellos es una lista de números enteros. Los nodos de esta lista deberán desenlazarse y reenlazarse en otras dos listas: una de números pares y otra de números impares. Al finalizar la función, la lista original deberá estar vacía y las otras dos listas deberán contener los nodos de acuerdo a si los números son pares o impares. La estructura de nodo para las tres listas es:

```
struct nodo {
   int dato;
   struct nodo *sig;
};
```

Dentro de la clase clase defina tres punteros a lista como datos privados.

8) Elabore una función que reciba como parámetro una lista simple y un número entero. Luego elimine de la lista todas las ocurrencias de ese número.

9) En una librería un cliente desea saber el número total de páginas de cierto libro, pero este se encuentra entre un montón de libros apilados sobre una mesa. Elabore una función recursiva que reciba la pila de libros y el título del libro, y devuelva la cantidad de páginas del mismo. Utilice la siguiente estructura:

```
struct libro{
char titulo[35];
   int numPaginas;
   libro *sig;
};
```