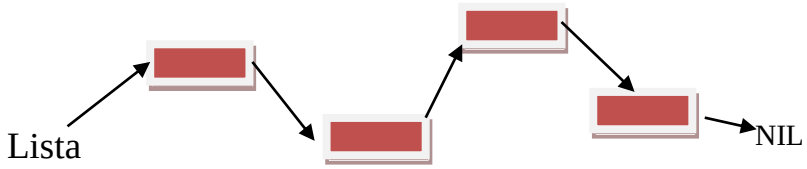
PRÁCTICA 9

LISTAS SIMPLES y DOBLES



Aclaración: los ejercicios marcados con * se recomiendan realizar en forma obligatoria durante la semana correspondiente a la realización de la práctica, acorde a lo estipulado en el cronograma. Además, se recomienda consultar la solución realizada con los ayudantes durante la práctica y de ser posible, escribir el programa en Lazarus Pascal y probar su ejecución. El resto de los ejercicios es necesario realizarlos como parte del estudio y preparación para el parcial.

Objetivos de la práctica:

Se espera que el alumno logre:

- **Utilizar la estructura de datos lista simplemente enlazada en la resolución de problemas tipo.**
- **Aplicar las operaciones de lista simples tales como agregar, insertar, borrar elementos de estas estructuras.**
- **Integrar conocimientos previos para la creación de soluciones a problemas del mundo real.**
- **Utilizar la estructura de datos lista doblemente enlazada en la resolución de problemas tipo.**
- **Aplicar las operaciones de lista dobles tales como agregar, insertar, borrar elementos de estas estructuras.**

Nota para todos los ejercicios: Modularizar las soluciones y definir las estructuras de datos utilizadas. Considerar que las listas pueden estar vacías y NO OLVIDE liberar la memoria ocupada antes de finalizar el programa.

1. ***Escribir un programa que lea una secuencia de números enteros terminada en 999 y los almacene en una lista simple. Utilizando la lista creada implementar:
   a) un módulo que reciba la lista y devuelva como resultado la cantidad de números que tienen exactamente 3 dígitos.
   b) un módulo que reciba la lista y un número y determine si dicho número está o no en la lista. La búsqueda debe terminar al encontrar la primera ocurrencia del número buscado.
   c) Un módulo que reciba la lista y libere la memoria reservada.

2. Escribir un programa que lea una secuencia de números enteros y genere una lista. La lectura finaliza cuando se lee el número 0. Una vez generada la lista, recorrerla una sola vez e informar:
   a) El mayor número leído.
   b) La cantidad de números cuya cantidad de dígitos es impar.
   c) Los 2 últimos números pares de la lista.

3. * Dada una lista de libros, donde cada libro está identificado por su **título** y su **autor**, definir una estructura de datos adecuada para almacenarlos. Escribir un programa que implemente los siguientes módulos:
   a) Calcular la longitud de la lista de libros.
   b) Calcular la cantidad de veces que aparece un autor dado (un autor puede haber escrito varios libros).
   c) Dado un autor, si existe, generar una nueva lista con los títulos de los libros que ha escrito.
   d) Agregar al final de la lista creada en **c)** un nuevo título de libro para ese autor.

4. * Dada una lista de aviones (marca, modelo y cantidad de asientos), definir la estructura que permita almacenarlos y escribir un programa que implemente:
   a) Un módulo que reciba una lista de aviones y un avión y lo agregue a la lista. Se sabe que la lista está ordenada por marca en forma descendente y se pide agregar el avión a la lista de forma que se mantenga el orden.
   b) Un módulo que reciba una lista de aviones y un avión, y elimine el elemento de la lista que coincida totalmente con el avión recibido. Además, debe retornar si la eliminación se realizó o no.

5. Un congreso de Informática ***dispone*** de una lista con la información de sus participantes. De cada participante, se conoce: Apellido y Nombre, año de nacimiento, Área de Investigación a la que se dedica y País de procedencia.
Se desea procesar la información para:
- **a)** Informar los nombres de las participantes menores de 30 años, con país de procedencia "Argentina" y cuya Área de Investigación es "Accesibilidad Web". Además, la cantidad total de participantes que cumplen con tal condición.
- **b)** Calcular e informar el porcentaje de participantes que provienen de países que no son "Argentina".
- **c)** Generar una nueva lista ordenada por Área de Investigación.

6. *Un cine ***dispone*** de una lista con la información de los tickets del día (Nro. Caja, nro. ticket, monto) correspondientes a sus 6 cajas. Escribir un programa que procese dicha lista y permita:
- **a)** Genere una nueva lista con los tickets con montos menores a $1000.
- **b)** Calcular e informar el monto promedio recaudado por caja.
- **c)** Calcular que cajas que recaudaron más de $12000 con menos de 100 tickets. Luego vuelva a recorrer la lista para generar una nueva con los tickets pertenecientes a las cajas que cumplan dicha condición. Recuerde usar un conjunto para mejorar la eficiencia.

7. *Un banco ***dispone*** de una lista con la información de pago de sus jubilados (Documento, apellido, nombre y monto a pagar). Debido a que la cantidad de jubilados para cobrar es muy grande se decidió pagarles en 10 días diferentes agrupándolos por el último dígito de su documento. Escribir un programa que implemente la separación de la lista en 10 listas diferentes conservando el ***orden original*** en cada lista.

8. Una empresa mantiene una lista de empleados. De un empleado se conoce su código de empleado, apellido y nombre, profesión, código de departamento al que pertenece, sueldo básico y antigüedad. Dicha lista está ordenada por código de empleado. Suponiendo que la lista ya existe se pide:
- **a)** Realizar un módulo que reciba un nuevo empleado y lo incorpore a la lista de empleados de la empresa (manteniendo el orden).
- **b)** Implementar un módulo que elimine de la lista todos los empleados que pertenezcan al departamento 4 o al departamento 10 (estos departamentos pueden no existir).

9. * Se cuenta con una lista que contiene información de las ventas realizadas por una empresa de venta de pasajes aéreos. Cada venta está compuesta por un nombre de persona, código de vuelo, categoría de pasaje y número de asiento. La lista puede contener 0, 1 o más registros por cada código de vuelo, y **está ordenada** por este campo.
El costo de un pasaje depende del vuelo y de su categoría. Se ***dispone*** de una estructura ***eficiente*** que por cada combinación de vuelo (son 30) y categoría (4 por vuelo) se almacena su precio.
- **a)** Generar una lista de registros que contenga por cada código de vuelo, el total de pasajes vendidos y el monto total recaudado.
- **b)** Calcular e informar los códigos de los 5 vuelos más vendidos.
- **c)** Generar una lista de los códigos de vuelos cuya cantidad de pasajes vendidos sea mayor que 46. La lista debe ir generándose ordenada por monto total a medida que se realiza el proceso a).

10.* Una casa de venta de comidas ofrece a la venta una variedad de platos. La misma cuenta con un software que utiliza dos estructuras de datos: una en donde almacena la información de sus clientes: código de cliente, apellido, nombre, domicilio y teléfono; y otra en donde almacena la información de los 150 platos diferentes que dispone. Los platos se encuentran codificados del 1 al 150 y de cada uno se sabe: nombre y stock disponible. Realizar un programa que:
- **a)** Lea información de pedidos de platos de comida. De cada pedido se conoce el código de cliente que desea hacer el pedido, el código del plato que solicita y la cantidad de ese plato. Para que el pedido pueda llevarse a cabo se deben realizar las validaciones correspondientes y en caso de no existir ningún inconveniente, realizar la venta del mismo, caso contrario, informar porque la venta no pudo ser realizada. Tener en cuenta que el código de cliente debe ser de un cliente almacenado por la empresa, y debe existir suficiente stock del plato pedido. La recepción de pedidos finaliza cuando se lee un código de cliente igual a -1.
- **b)** Una vez finalizada la recepción de solicitudes informar aquellos platos que ya no tienen más stock disponible.

11. *Un deportólogo está realizando una investigación sobre el rendimiento de los maratonistas. Se seleccionan 50 maratonistas para ser estudiados. De cada uno se conoce el nombre, apellido, género, y el tiempo (minutos y segundos) registrado en cada maratón que ha corrido. El deportólogo necesita un programa para:

a) Cargar la información de los 50 maratonistas. Para esto puede seleccionar el criterio que desee para cargar el tiempo de las maratones.
b) Informar para cada maratonista, la maratón con el mejor tiempo de desempeño.
c) Calcular el promedio de tiempo de cada maratonista.

12. * Realice dos módulos que reciban dos listas ordenadas de enteros A y B y devuelvan:
a) En el primer módulo: otra lista C ordenada que sea la resultante de unir las dos listas recibidas.
b) En el segundo módulo: la lista A donde se inserten los elementos de la lista B, y la lista A continúa ordenada.
c) Realice además un tercer módulo que reciba la lista A, y un número entero, y se eliminen todas las ocurrencias de ese número de la lista.

13. * Defina un tipo de datos adecuado para una lista doblemente enlazada de enteros e implemente:
a) módulos para: agregar adelante, agregar atrás, insertar ordenado.
b) un módulo que reciba la lista y retorne si la lista contiene la misma secuencia de números de derecha a izquierda que de izquierda a derecha (si es capicúa).

14. Escribir un programa que lea una secuencia de números enteros terminada en 999 y los almacene en una lista doblemente enlazada. Utilizando la lista creada implementar un módulo que reciba la lista y devuelva como resultado la cantidad de números con 3 dígitos.

15. Escribir un módulo que recibe una lista doblemente enlazada de caracteres y retorne si la palabra almacenada es palíndrome, es decir que contiene la misma secuencia de letras de derecha a izquierda que de izquierda a derecha.

16. *Analice los siguientes módulos que tienen como tarea agregar un nuevo nodo al inicio de una lista doblemente enlazada. Indique cuál de los cuatro módulos realiza de forma correcta la operación, y fundamente su respuesta. Considere las siguientes declaraciones:

```
Type
  ListaDE =^nodo;
  nodo = record
    dato:integer;
     ant: ListaDE;
     sig: ListaDE;
  end;
  Punteros = record
       Pri1, Pri2: ListaDE;
  end;
Var p: punteros;
```

caso 1:

```
Procedure agregarAlInicio(var p:punteros; num:integer);
Var nue: ListaDE;
Begin
  New(nue);
  nue^.dato:=num;
  nue^.sig := p.pri1;
  nue^.ant := nil;
  if (p.pri1 <> nil) then
      p.pri1^.ant := nue;
  else
      p.pri1 := nue;
 p.pri2 := nue;
end;
```

caso 2:

```
Procedure agregarAlInicio(var p:punteros; num:integer);
Var nue: ListaDE;
Begin
   New(nue);
   nue^.dato:=num;
   nue^.sig := p.pri1;
   nue^.ant := nil;
   if (p.pri1 <> nil) then
     p.pri1^.sig := nue;
  else
     p.pri2 := nue;
  p.pri2:= nue;
 end;
```

caso 3:

```
Procedure agregarAlInicio (var p:punteros; num:integer);
Var nue: ListaDE;
Begin
    New(nue);
    nue^.dato:=num;
    nue^.sig := p.pri2;
    nue^.ant := nil;
    if (p.pri2 <> nil) then
          p.pri1^.sig := nue;
    else
         p.pri2 := nue;
   p.pri1 := nue;
end;
```

caso 4:

```
Procedure agregarAlInicio(var p:punteros; num:integer);
Var nue: ListaDE;
Begin
   New(nue);
   nue^.dato:=num;
   nue^.sig := p.pri1;
   nue^.ant := nil;
   if (p.pri1 <> nil) then
        p.pri1^.ant := nue;
   else
        p.pri2 := nue;
   p.pri1 := nue;
end;
```