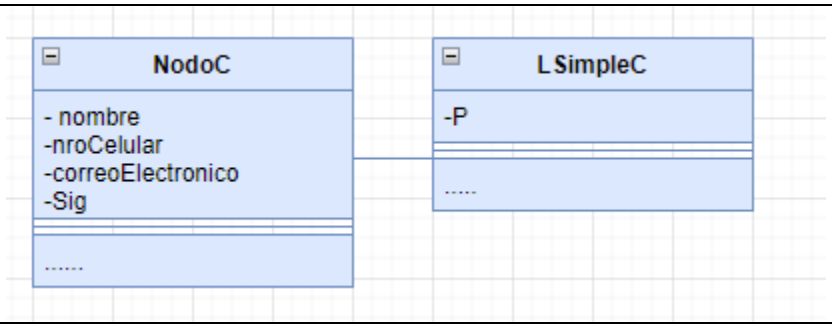
Materia: **LAB - 131**	**Paralelo:** **“D”**
Docente: Lic. Carmen Rosa Huanca Quisbert	Carrera: Informática
Nombres y Apellidos: **Yhorel Yhared Alvarez Alvarez**	
CI: 12865468 LP	**Fecha:** **29/04/2022**
Tema: **Listas Simples Dobles Simples circulares y dobles circulares**	

1. Crear una lista simple normal de contactos <nombre, nrocelular, correoElectronico>.

    a. Agregar un nuevo contacto, si el número ya está registrado no agregar y mostrar un mensaje.
    b. Mostrar el nombre del contacto con correo electrónico X.
    c. Eliminar al contacto con nombre X



CÓDIGO JAVA

```
public static void main(String[] args) {
    LSimpleC a = new LSimpleC();
    a.adifinal("nombre1", 123, "correo1");
    a.adifinal("nombre2", 973, "correo2");
    a.adifinal("nombre3", 537, "correo3");
    a.adifinal("nombre4", 321, "correo4");
    a.adifinal("nombre5", 951, "correo5");
    a.adifinal("nombre6", 357, "correo6");
    a.adifinal("nombre7", 654, "correo7");
    a.adifinal("nombre8", 456, "correo8");
    a.mostrar();

    // Solucion A
    System.out.println("\nSolucion A");
    SolA(a,"nombre",123,"correo");
    System.out.println("\nSolucion B");
    SolB(a,"correo3");
    System.out.println("\nSolucion C");
    SolC(a,"nombre8");
```

```
    a.mostrar();
  }
```

Inciso a

```
public static void SolA(LSimpleC a, String nm, int n, String cc){
    NodoC x = a.getP();
    boolean sw = true;
    while(x != null){
      if(x.getNroCelular() == n){
        sw = false;
      }
      x = x.getSig();
    }
    if(sw){
      a.adifinal(nm, n, cc);
      a.mostrar();
    }
    else{
      System.out.println("ya existe el numero");
    }
  }
```

Inciso b

```
public static void SolB(LSimpleC a, String x){
    NodoC nc = a.getP();
    while(nc != null){
      if(nc.getCorreoElect().equals(x)){
        System.out.println(nc.getNombre());
      }
      nc = nc.getSig();
    }
  }
```

Inciso c

```
public static void SolC(LSimpleC a, String x){
    NodoC nc = a.getP();
    NodoC ant = new NodoC();
    int c = 0;
    while(nc != null){
      if(x.equals(nc.getNombre())){
        if(c != 0){
          ant.setSig(nc.getSig());
        }
        else{
          a.setP(nc.getSig());
        }
      }
      c++;
      ant = nc;
      nc = nc.getSig();
    }
```

CORRIDA

```
nombre1 123 correo1
nombre2 973 correo2
nombre3 537 correo3
nombre4 321 correo4
nombre5 951 correo5
nombre6 357 correo6
nombre7 654 correo7
nombre8 456 correo8
```

Inciso a

```
Solucion A
ya existe el numero
```

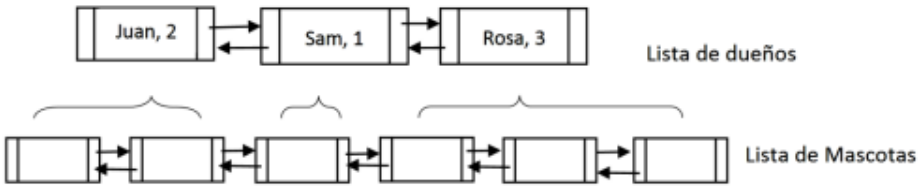
Inciso b

```
Solucion B
nombre3
```

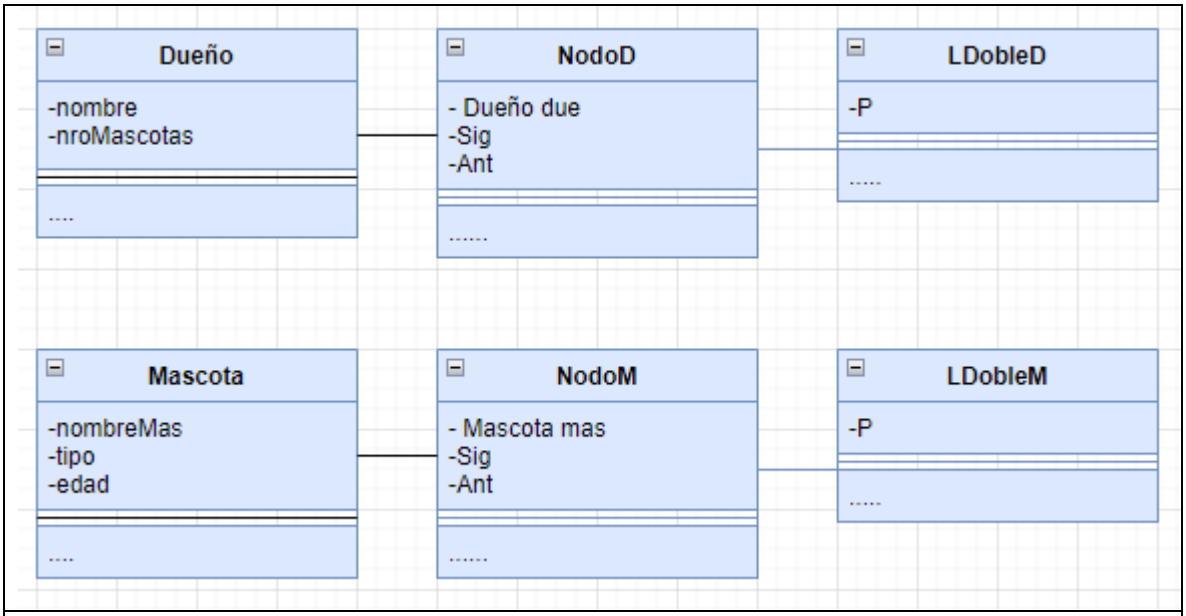
Inciso c

```
Solucion C
nombre1 123 correo1
nombre2 973 correo2
nombre3 537 correo3
nombre4 321 correo4
nombre5 951 correo5
nombre6 357 correo6
nombre7 654 correo7
```

2. En el registro para la atención en una veterinaria, se tiene una lista doble normal de objetos dueño<nombre, nroMascotas> y otra lista doble normal de mascotas<nombreMas, tipo, edad>



a. Mostrar las mascotas del dueño con nombre X.
b. Adicionar una nueva mascota al dueño con nombre X.
c. Eliminar al(los) dueño(s) y sus mascotas que tenga(n) a una(s) "tortuga" de mascota(s).



CÓDIGO JAVA

```
public static void main(String[] args) {
	Dueño d1 =  new Dueño("nombre1", 2);
	Dueño d2 =  new Dueño("nombre2", 3);
	Dueño d3 =  new Dueño("nombre3", 2);
	Dueño d4 =  new Dueño("nombre4", 1);
	LDobleD l1 = new LDobleD();
	l1.adifinal(d1);
	l1.adifinal(d2);
	l1.adifinal(d3);
	l1.adifinal(d4);

	Mascota m1 = new Mascota("mascota1", "tortuga", 10);
	Mascota m2 = new Mascota("mascota2", "perro", 5);
	Mascota m3 = new Mascota("mascota3", "gato", 8);
	Mascota m4 = new Mascota("mascota4", "gato", 3);
	Mascota m5 = new Mascota("mascota5", "peroo", 1);
	Mascota m6 = new Mascota("mascota6", "perico", 9);
	Mascota m7 = new Mascota("mascota7", "loro", 4);
	Mascota m8 = new Mascota("mascota8", "gato", 7);

	LDobleM l2 = new LDobleM();
	l2.adifinal(m1);
	l2.adifinal(m2);
	l2.adifinal(m3);
	l2.adifinal(m4);
	l2.adifinal(m5);
	l2.adifinal(m6);
	l2.adifinal(m7);
	l2.adifinal(m8);
```

```
      System.out.println("\n -------- Solucion A -------- ");
      SolA(l1, l2, "nombre4");
      System.out.println("\n -------- Solucion B -------- ");
      Mascota mm = new Mascota("nuevo", "perro", 10);
      SolB(l1, l2, "nombre3",mm);
      mostrar(l1,l2);
      System.out.println("\n -------- Solucion C -------- ");
      SolC(l1, l2);
      mostrar(l1,l2);
}
public static void mostrar(LDobleD a, LDobleM b){
      NodoD nd = a.getP();
      NodoM nm = b.getP();
      while(nd != null){
         nd.getDue().mostrar();
         for(int i=0; i<nd.getDue().getNroMascotas(); i++){
            System.out.println("\t"+nm.getMas().getNombreMas()+" "+nm.getMas().getTipo());
            nm = nm.getSig();
         }
         nd = nd.getSig();
      }
   }
```

Inciso a

```
public static void SolA(LDobleD a, LDobleM b, String x){
      NodoD nd = a.getP();
      NodoM nm = b.getP();
      while(nd != null){
         if(nd.getDue().getNombre().equals(x)){
            nd.getDue().mostrar();
            for(int i=0; i<nd.getDue().getNroMascotas(); i++){
               System.out.println("\t"+nm.getMas().getNombreMas()+" "+nm.getMas().getTipo());
               nm = nm.getSig();
            }
         }
         else{
            for(int i=0; i<nd.getDue().getNroMascotas(); i++){
               nm = nm.getSig();
            }
         }
         nd = nd.getSig();
      }
   }
```

Inciso b

```
public static void SolB(LDobleD a, LDobleM b, String x,Mascota ma){
      NodoD nd = a.getP();
      NodoM nm = b.getP();
      NodoM nue = new NodoM();
```

```
    nue.setMas(ma);
    while(nd != null){
      if(nd.getDue().getNombre().equals(x)){
        for(int i=0; i<nd.getDue().getNroMascotas(); i++){
          nm = nm.getSig();
        }
        NodoM aux = nm;
        nm = nm.getAnd();
        nm.setSig(nue);
        nue.setAnd(nm);
        nue.setSig(aux);
        nd.getDue().setNroMascotas(nd.getDue().getNroMascotas()+1);
      }
      else{
        for(int i=0; i<nd.getDue().getNroMascotas(); i++){
          nm = nm.getSig();
        }
      }
      nd = nd.getSig();
    }
  }
```

Inciso c

```
public static void SolC(LDobleD a, LDobleM b){
    NodoD nd = a.getP();
    NodoM nm = b.getP();
    int cont = 0;
    while(nd != null){
      NodoM mp = nm;
      NodoM mu = null;
      boolean sw = false;
      for(int i=0; i<nd.getDue().getNroMascotas()-1; i++){
        if(nm.getMas().getTipo().equals("tortuga")){
          sw = true;
        }
        nm = nm.getSig();
      }
      mu = nm;
      if(mu.getMas().getTipo().equals("tortuga")){
        sw = true;
      }
      if(sw){
        if(cont == 0){
          b.setP(mu.getSig());
          a.setP(nd.getSig());
        }
        else{
          mp.getAnd().setSig(mu.getSig());
          nd.getAnd().setSig(nd.getSig());
```

```
        }
      }
      else{
        nm = nm.getSig();
      }
      nd = nd.getSig();
      cont++;
    }
  }
```

CORRIDA

Inciso a

```
-------- Solucion A --------
< nombre4 1 >
        mascota8 gato
```

Inciso b

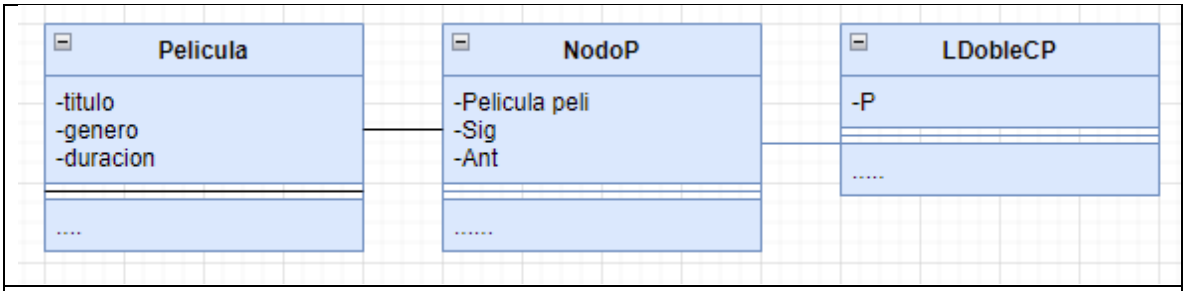
```
-------- Solucion B --------
< nombre1 2 >
        mascota1 tortuga
        mascota2 perro
< nombre2 3 >
        mascota3 gato
        mascota4 gato
        mascota5 peroo
< nombre3 3 >
        mascota6 perico
        mascota7 loro
        nuevo perro
< nombre4 1 >
        mascota8 gato
```

Inciso c

```
-------- Solucion C --------
< nombre2 3 >
        mascota3 gato
        mascota4 gato
        mascota5 peroo
< nombre3 3 >
        mascota6 perico
        mascota7 loro
        nuevo perro
< nombre4 1 >
        mascota8 gato
```

3. Sea una lista Doble Circular de Películas <título, genero, duración> se pide:

   a. Llevar los k primeros nodos al final de la lista
   b. Eliminar a la(s) película(s) que tiene(n) menos duración.
   c. Agregar una nueva película antes de la última película de genero X



CÓDIGO JAVA

```
public static void main(String[] args) {
    Pelicula p1 = new Pelicula("titulo1", "accion", 120);
    Pelicula p2 = new Pelicula("titulo2", "accion", 130);
    Pelicula p3 = new Pelicula("titulo3", "terror", 100);
    Pelicula p4 = new Pelicula("titulo4", "infaltil", 90);
    Pelicula p5 = new Pelicula("titulo5", "accion", 100);
    Pelicula p6 = new Pelicula("titulo6", "ficcion", 120);
    Pelicula p7 = new Pelicula("titulo7", "accion", 90);
    Pelicula p8 = new Pelicula("titulo8", "ficcion", 110);
    Pelicula p9 = new Pelicula("titulo9", "drama", 115);

    LDobleCD ll = new LDobleCD();
    ll.adifinal(p1);
    ll.adifinal(p2);
    ll.adifinal(p3);
    ll.adifinal(p4);
    ll.adifinal(p5);
    ll.adifinal(p6);
    ll.adifinal(p7);
    ll.adifinal(p8);
    ll.adifinal(p9);
    ll.mostrar();
    System.out.println("\n Inciso A ");
    SolA(ll,5);
    ll.mostrar();
    System.out.println("\n Inciso B ");
    SolB(ll);
    ll.mostrar();
    System.out.println("\n Inciso C ");
    Pelicula pp = new Pelicula("nuevo", "Romantisismo", 115);
    SolC(ll, pp, "drama");
    ll.mostrar();
```

```
   }
```

Inciso a

```
public static void SolA(LDobleCD a, int k){
      for(int i=0; i<k; i++){
         NodoP r = a.getP();
         NodoP aux = r.getSig();
         a.setP(aux);
      }
   }
```

Inciso b

```
public static void SolB(LDobleCD a){
      NodoP r = a.getP();
      int mn = r.getDue().getDuracion();
      while (r.getSig() != a.getP()){
         if(r.getDue().getDuracion() < mn){
            mn = r.getDue().getDuracion();
         }
         r = r.getSig();
      }
      if(r.getDue().getDuracion() < mn){
         mn = r.getDue().getDuracion();
      }
      r = a.getP();
      while (r.getSig() != a.getP()){
         if(r.getDue().getDuracion() == mn){
            r.getSig().setAnd(r.getAnd());
            r.getAnd().setSig(r.getSig());
         }
         r = r.getSig();
      }
      if(r.getDue().getDuracion() == mn){
         r.getSig().setAnd(r.getAnd());
         r.getAnd().setSig(r.getSig());
      }
}
```

Inciso c

```
public static void SolC(LDobleCD a, Pelicula pe, String x){
      NodoP r = a.getP();
      int c = 0;
      int p = 0;
      while (r.getSig() != a.getP()){
         if(r.getDue().getGenero().equals(x)){
            p = c;
         }
         r = r.getSig();
         c++;
      }
      if(r.getDue().getGenero().equals(x)){
```

```
        p = c;
    }
    c++;
    r = a.getP();
    c = 0;
    while (r.getSig() != a.getP()){
        if(c == p){
            NodoP nue = new NodoP();
            nue.setDue(pe);
            r.getAnd().setSig(nue);
            nue.setAnd(r.getAnd());
            nue.setSig(r);
            r.setAnd(nue);
        }
        r = r.getSig();
        c++;
    }
    if(c == p){
        NodoP nue = new NodoP();
        nue.setDue(pe);
        r.getAnd().setSig(nue);
        nue.setAnd(r.getAnd());
        nue.setSig(r);
        r.setAnd(nue);
    }
    c++;
}
```

CORRIDA

```
run:
< titulo1 accion 120 >
< titulo2 accion 130 >
< titulo3 terror 100 >
< titulo4 infaltil 90 >
< titulo5 accion 100 >
< titulo6 ficcion 120 >
< titulo7 accion 90 >
< titulo8 ficcion 110 >
< titulo9 drama 115 >
```

Inciso a

```
 Inciso A
< titulo6 ficcion 120 >
< titulo7 accion 90 >
< titulo8 ficcion 110 >
< titulo9 drama 115 >
< titulo1 accion 120 >
< titulo2 accion 130 >
< titulo3 terror 100 >
< titulo4 infaltil 90 >
< titulo5 accion 100 >
```

Inciso b

```
  Inciso B
 < titulo6 ficcion 120 >
 < titulo8 ficcion 110 >
 < titulo9 drama 115 >
 < titulo1 accion 120 >
 < titulo2 accion 130 >
 < titulo3 terror 100 >
 < titulo5 accion 100 >
```

Inciso c

```
 Inciso C
< titulo6 ficcion 120 >
< titulo8 ficcion 110 >
< nuevo Romantisismo 115 >
< titulo9 drama 115 >
< titulo1 accion 120 >
< titulo2 accion 130 >
< titulo3 terror 100 >
< titulo5 accion 100 >
```