1) En una librería tienen pretenden cargar en una lista los datos de todos los libros que se encuentran a la venta.
Por cada libro se tiene la siguiente información: código (es un código que permite identificar a cada libro), titulo, autor, cantidad de páginas y precio.
Construya un programa que permita generar esa lista y los siguientes métodos.

a) Un método que permita agregar un nuevo libro a la lista.
b) Un método que recibiendo un código de libro pueda eliminarlo de la lista.
c) Un método que retorne la cantidad de libros de la lista.
d) Un método que retorne la cantidad de libros que tienen mas de X páginas, siendo X un parámetro.
e) Un método que retorne el código de libro de mayor cantidad de páginas.
f) Un método que retorne verdadero si se encuentra un determinado libro, recibiendo por parámetro el título.

2) En un observatorio ambiental realizarán un experimento para lo cual tomaran 10 medidas diarias del nivel de monóxido de carbono durante una cantidad no determinada de dias.
La información correspondiente a cada día será almacenada en una lista para que luego, recorriéndola, se pueda obtener la siguiente información:

a) Cual es el promedio de contaminación de un determinado día. Para esto se recibe por parámetro el número de día. El nivel promedio de un día se obtiene a partir de sus 10 mediciones.
b) Cual es el número de día de mayor nivel promedio de contaminación.
c) Cual es el nivel medio de contaminación de todos los días de la lista.
d) Un método que retorne verdadero si en la lista hay días con alguna medición mayor a cierto parámetro.
e) Un método que permita agregar un nuevo día a la lista.
f) Un método que reciba por parámetro un número de día y permita aliminarlo de la lista.

3) Construir una lista de alumnos de manera que cada nodo represente a un alumno en particular. Por cada alumno se tiene esta información: nombre, legajo y un vector de 5 notas que representan las calificaciones obtenidas en la presentación de cinco trabajos prácticos para una determinada materia.
A partir de la lista responda los siguientes puntos:

a) ¿Cual es el legajo del aluno con mayor promedio?.
b) ¿Como se llama el aluno de menor promedio?.
c) ¿Existen alumnos que no hayan presentado ningún trabajo practico?.
d) Conociendo el legajo, ¿Cual es promedio general de ese alumno?.
e) ¿Cual es el promedio general de todos los alumnos de la lista?.

4) En una empresa dedicada a la realización de mediciones de audiencia de programas de televisión, pretenden construir un software que permita cargar en una lista los datos de las mediciones realizadas para cada programa. Estos datos son: Código de programa (numérico), puntos de raiting, fecha en que se tomo la lectura (tres números enteros, uno para el día otro para el mes y otro para el alño).
Una vez que la lista se halla cargado, se debe poder obtener la siguiente información:

a) Cual es el código programa que ha registrado la mayor medición.
b) En que día se ha registrado la menor medición.
c) Tomando por parámetro un código de programa, retornar el promedio de sus mediciones.
d) Tomando por parámetro una fecha y un código de programa, retornar verdadero si hay mediciones con esos datos y falso si no las hay.
e) Tomando por parámetro un número de mes, retornar la cantidad de mediciones de ese mes.

5) Tomando el mismo programa generado en el punto anterior, cargue ahora en un vector el código de programa y su nombre correspondiente y realice las siguientes tareas:

a) Tomando por parámetro el código de un programa, mostrar el nombre correspondiente.
b) Mostrar en la pantalla el nombre de programa que ha tenido la mayor medición.
c) Tomando por parámetro el nombre de un programa, mostrar el promedio de sus mediciones.

d) Mostrar por pantalla el nombre de cada programa junto al promedio de sus mediciones.

6) Suponga que debemos programar una pequeña parte de un administrador de impresiones. Este software debe almacenar en una cola de impresión todos los trabajos pendientes para poder mandarlos a la impresora cuando sea posible. Por cada trabajo a imprimir se tiene simplemente un "string" con los caracteres que deben ser puestos en la impresora.
Nosotros debemos programar solo la clase "trabajo" que representa los trabajos a imprimir y la clase "colaDeImpresión" que administra la cola de trabajos pendientes de imprimir.
Esta clase debe ser capaz de realizar las siguientes tareas.

a) Agregar un nuevo trabajo a la cola de impresión.
b) Quitar un trabajo de la cola de impresión.
c) Retornar la cantidad de trabajos pendientes de imprimir.
d) Vaciar toda la cola de impresión.
e) Retornar el trabajo que se encuentra primero en la cola de impresión.
f) Retornar el trabajo que se encuentra al último en la cola de impresión.

7) En un banco están construyendo un programa que les permita representar la cola de personas que se ordena frente de una caja para poder realizar pagos de las facturas de sus servicios.
Por cada integrante que se encuentra en la cola se almacena como única información la hora de llegado (en un número entero).
Construir una clase que permita representar esta cola de espera y realice las siguientes actividades.

a) Un método que permita agregar un nuevo elemento a la cola recibiendo por parámetro su hora de llegada.
b) Un método que, recibiendo por parámetro la hora en la que ha sido atendida una persona en la caja, pueda quitarla de la cola de espera retornando el tiempo total de espera.
c) Un método que, recibiendo por parámetro la hora actual, retorne el tiempo promedio de espera general de todas las personas que se encuentran en la cola.
d) Un método que retorne verdadero si existen en la cola personas que hallan llegado a la misma hora.

8) En una fabrica existe una máquina que permite realizar cierto producto de manera totalmente automatizada. En un extremo de dicha máquina se encuentra una cinta transportadora que arroja los productos terminado a un recipiente donde se almacenan en forma de pila ya que los nuevos van cayendo sobre los fabricados previamente.
Un operario retira de a uno los elementos de esta pila para poder llevarlos al almacén de productos terminados.
Nosotros necesitamos construir un programa que nos permita representar esta pila de productos sabiendo que por cada uno se debe almacenar la siguiente información: Código del producto, peso y longitud. El mismo programa debe ser capaz de:

a) Agregar un nuevo producto a la pila para representar el momento en que la maquina finaliza la elaboración de un nuevo elemento.
b) Quitar un producto de la pila para representar el momento en que el operario retira un elemento retornando su código.
c) Retornar la longitud del elemento que esta primero para ser quitado.
d) Retornar verdadero si no hay elementos en la pila.
e) Retornar la longitud promedio de los elementos de la pila.

9) Tenemos que construir un programa que nos permita representar el siguiente juego de cartas:
Las cartas apilan formando un mazo que se coloca boca abajo para que dos jugadores saquen de a un naipe siempre desde la parte superior.
Cada vez que un jugador obtiene una carta con un número mayor al que ha retirado anteriormente puede seguir sacando otra carta. Si rompe la secuencia, es decir, obtiene una igual o menor a la ultima que ha quitado del mazo, debe ceder el turno al segundo jugador.

El juego finaliza cuando ya no hay cartas en el mazo y de los dos jugadores, gana quien tenga mas cartas en su poder.
Para construir el programa, note que el mazo de cartas puede ser representado por una pila que se llena por única vez cuando se colocan todas las cartas en el mazo y donde se quita de a una carta cuando el jugador realiza su movimiento. Por cada carta se guarda solo su número.
El juego en si puede ser representado por la clase "juego" que debe tener una lista en su interior y poder controlar la cantidad de cartas retiradas por cada jugador para poder informar quien es el ganador al quedar el mazo vacío.

10) Realizar un programa que permita cargar en una lista los datos de las encomiendas transportadas por una empresa de correo. Cada encomienda que la empresa transporta deberá estar representada por un nodo de la lista que contenga: el código que identifica a la encomienda, el peso del paquete, el monto asegurado, el código de la localidad de origen y el código de la localidad de destino.
Realice las siguientes actividades.

a) Un método que permita agregar una nueva encomienda a la lista.
b) Un método que, tomando como parámetro el código de una encomienda, pueda quitarla de la lista.
c) Un método que, tomando por parámetro un peso, retorne la cantidad de paquetes que son mas pesados que dicho valor.
d) Un método que retorne el monto asegurado promedio de todas las encomiendas de la lista.
e) Un método que, recibiendo por parámetro un código de destino, cargue en un vector las encomiendas que se encuentran es la lista y que serán entregadas en dicho destino.

11) Con el mismo caso del punto anterior, construya ahora una clase que permita cargar un vector con 10 códigos de localidades y su correspondiente descripción, y obtenga la siguiente información:

a) Dado un número de encomienda, mostrar la descripción de su origen y su destino.
b) Dado un nombre de origen, retornar la cantidad de encomiendas que ha partido desde allí.
c) Hacer un método que retorne verdadero si hay encomiendas para entregar en un determina destino, recibiendo por parámetro el nombre de la localidad.
d) Cargar en una matriz, que represente en sus filas los 10 orígenes y en sus columnas los 10 destinos las cantidades de encomiendas que se encuentran en la lista para cada caso. Por ejemplo, si en la lista existen 5 nodos que tienen como origen la localidad 3 y como destino a la 8, entonces en la intersección fila 3 columna 8 deberá colocar un número 5.
e) Recorriendo la matriz generada en el punto anterior, mo strar las cantidades de cada combinación de origen y destino acompañadas del nombre de cada localidad.

Solución Ej2 sin herencia

Esta solución esta planteada a partir de una clase "libro" que modela el concepto del libro de la librería.
Luego la clase "nodo" representa al nodo de la lista de manera que existe una relación del tipo "tiene un" ya que un nodo tienen en su interior a un libro.
Finalmente la clase "listaLibros" modela todo el comportamiento de la lista.

```
import java.io.*;
import utn.frc.io.In;

class libro  // clase para representar los libros de la lista
{
        private int codigo, cantPaginas;
        private String titulo, autor;
        private float precio;

        // cosntructor con parámetros
```

```
	public libro(int cod, int cant, String tit, String aut, float pre)
	{
		codigo = cod;
		cantPaginas = cant;
		titulo = tit;
		autor = aut;
		precio = pre;
	}

	// cosntructor sin parámetros
	public libro()
	{
		this(0,0,"","",0);
	}


	// metodo que retorna el código del libro
	public int getCodigo()
	{
		return codigo;
	}

	// metodo que retorna la cantidad de paginas
	public int getPaginas()
	{
		return cantPaginas;
	}

	// metodo que retorna el titulo de un libro
	public String getTitulo()
	{
		return titulo;
	}

	// metodo que tostring para visualizar los datos del libro
	public String toString()
	{
		String retorno;
		retorno = "Codigo: " + codigo + ", Páginas: " +
		cantPaginas +", Título: "+ titulo + ", Autor: " +
		autor + ", Precio: " + precio;
		return retorno;
	}

}  // fin de la clase libro


// clase nodo, contiene un libro y una referencia al nodo siguiente
class nodo
{
	private libro l;
	private nodo proximo;

	// constructor con parámetros
	public nodo(int cod, int cant, String tit, String aut, float pre)
	{
```

```
                l = new libro(cod, cant, tit, aut, pre);
                proximo = null;
        }

        // cosntructor sin parámetros
        public nodo()
        {
                l = new libro(0,0,"","",0);
                proximo = null;
        }

        // metodo que retorna el libro del nodo
        public libro getLibro()
        {
                return l;
        }

        //metodo que retorna el codigo del libro del nodo
        public int getCodigo()
        {
                return l.getCodigo();
        }

        // metodo que retorna la cantidad de paginas del libro del nodo
        public int getPaginas()
        {
                return l.getPaginas();
        }

        // metodo que retorna la referencia el proximo del nodo

        // metodo que retorna el titulo del nodo del libro
        public String getTitulo()
        {
                return l.getTitulo();
        }
        public nodo getProximo()
        {
                return proximo;
        }

        // metodo que permite setar el puntero al proximo
        public void setProximo(nodo prox)
        {
                proximo = prox;
        }

        // meto toStrin para visualizar los datos del nodo...
        public String toString()
        {
                return l.toString();
        }

} // fin de la clase nodo
```

```
// clase que representa la lista de libros
public class listaLibros
{
	private nodo primero;

	// metodo quepermite agregar un libro. Utiliza insercion al frente.
	// punto a) del ejercicio uno
	public void agregar(int cod, int cant, String tit, String aut, float pre)
	{
		nodo aux;
		aux = primero;
		primero = new nodo(cod, cant, tit, aut, pre);
		primero.setProximo(aux);
	}

	// metodo que tomando un código de libro pueda eliminarlo de la lista
	// punto b) del ejercicio uno
	public void eliminar(int cod)
	{
		nodo aux1, aux2, aux3;
		// veamos si el que hay que eliminar es el primero
		if(primero.getCodigo()==cod)
		{
			aux1 = primero.getProximo();
			primero.setProximo(aux1);
		}
		else // no es el primero, ahy que buscar antes de borrar
		{
			aux1 = primero;
			for(aux2 = primero.getProximo();aux2 != null && aux2.getCodigo() != cod;
aux2=aux2.getProximo())
			{
				aux1 = aux2;
			}
			if(aux2 != null) // encontro el nodo buscado!!!
			{
				aux3 = aux2.getProximo();
				aux1.setProximo(aux3);
			}
		}
	}

	// metodo qeu retorna la cantidad de libros de la lista
	// punto c) del ejercicio uno
	public int getCantidad()
	{
		int cant = 0;
		nodo aux1;
		for(aux1 = primero; aux1 != null; aux1=aux1.getProximo())
		{
			cant ++;
		}
		return cant;
	}

	// metodo que retorna la cantidad de libros con mas de X paginas
```

```
// punto d) ejercicio uno
public int getCantidadMasPaginas(int cant)
{
	int cantidad = 0;
	nodo aux1;
	for(aux1 = primero; aux1 != null; aux1=aux1.getProximo())
	{
		if(aux1.getPaginas() > cant)  // es de los que buscamos....
		  cantidad ++;
	}
	return cantidad;
}

// metodo que retorna el código del libro de mayor cantidad de pagians
// punto e) del ejerciocio uno
public int getCodigoMasPaginas()
{
	int cant, cod;
	nodo aux1;
	// tomamos como valores iniciales los del primer nodo de la lista
	cant = primero.getPaginas();
	cod = primero.getCodigo();
	for(aux1 = primero; aux1 != null; aux1=aux1.getProximo())
	{
		if(aux1.getPaginas() > cant)  // tiene mas paginas que los otros
		  cod = aux1.getCodigo();
	}
	return cod;
}

// metodo que retorna verdadero si un libro esta en la lista
// punto d) del ejercicio uno.
public boolean estaElTitulo(String tit)
{
	boolean esta = false;
	nodo aux1;
	for(aux1 = primero; aux1 != null; aux1=aux1.getProximo())
	{
		if(aux1.getTitulo() == tit)  // es de los que buscamos....
		  esta = true;
	}
	return esta;
}


//meto toStrin de la lista como para poder ver que datos tiene...
public String toString()
{
	String retorno;
	retorno = "";
	nodo aux1;
	for(aux1 = primero; aux1 != null; aux1=aux1.getProximo())
	{
		retorno += aux1.toString() + "  -  ";
	}
	return retorno;
```

```
        }

} // fin de la clase listaLibros



// un pequeño programa como para ver algunos resultados...
class programa
{
 public static void main(String args[])
 {
        int resultado;
        boolean esta;
        listaLibros lista = new listaLibros();
        lista.agregar(10, 150, "Programacion en Java", "Jorge Tymoschuk", 150);
        lista.agregar(20, 90, "Base de datos", "Silvio Serra", 120);
        lista.agregar(25, 130, "Como aprobar AED", "Mandrake", 1000);
        lista.agregar(4, 35, "Estudie corte y confección", "Utilisima", 25);
        System.out.println(lista.toString());
        lista.eliminar(10);
        System.out.println(lista.toString());
        resultado = lista.getCantidadMasPaginas(100);
        System.out.println("La cantidad de libros con más de 100 páginas es: " + resultado);
        resultado = lista.getCodigoMasPaginas();
        System.out.println("El código del libro con más páginas es: " + resultado);
        esta = lista.estaElTitulo("programacion en java");
        if(esta = true)   // el libro si esta
    System.out.println("El libro Programacion en java si esta en la lista");
   else
    System.out.println("El libro Programacion en java no esta en la lista");
 }
}
```

Solución Ej1 con herencia.

Esta es una forma de resolver el ejercicio uno pero ahora haciendo que la clase “nodo” herede se “libro”. Desde este punto de viste, podríamos decir que el nodo “es un libro” que además tiene una referencia al nodo siguiente. Veamos….

```
import java.io.*;
import utn.frc.io.In;

class libro  // clase para representar los libros de la lista
{
        private int codigo, cantPaginas;
        private String titulo, autor;
        private float precio;

        // cosntructor con parámetros
        public libro(int cod, int cant, String tit, String aut, float pre)
        {
                codigo = cod;
                cantPaginas = cant;
                titulo = tit;
                autor = aut;
```

```
		precio = pre;
	}

	// cosntructor sin parámetros
	public libro()
	{
		this(0,0,"","",0);
	}


	// metodo que retorna el código del libro
	public int getCodigo()
	{
		return codigo;
	}

	// metodo que retorna la cantidad de paginas
	public int getPaginas()
	{
		return cantPaginas;
	}

	// metodo que retorna el titulo de un libro
	public String getTitulo()
	{
		return titulo;
	}

	// metodo que tostring para visualizar los datos del libro
	public String toString()
	{
		String retorno;
		retorno = "Codigo: " + codigo + ", Páginas: " +
		cantPaginas +", Título: "+ titulo + ", Autor: " +
		autor + ", Precio: " + precio;
		return retorno;
	}

}  // fin de la clase libro


// clase nodo, contiene un libro y una referencia al nodo siguiente
class nodo extends libro
{
	private nodo proximo;

	// constructor con parámetros
	public nodo(int cod, int cant, String tit, String aut, float pre)
	{
		super(cod, cant, tit, aut, pre);
		proximo = null;
	}

	// cosntructor sin parámetros
	public nodo()
	{
```

```
		super(0,0,"","",0);
		proximo = null;
	}

	// metodo que retorna la referencia el proximo del nodo
	public nodo getProximo()
	{
		return proximo;
	}

	// metodo que permite setar el puntero al proximo
	public void setProximo(nodo prox)
	{
		proximo = prox;
	}


} // fin de la clase nodo


// clase que representa la lista de libros
public class listaLibros
{
	private nodo primero;

	// metodo quepermite agregar un libro. Utiliza insercion al frente.
	// punto a) del ejercicio uno
	public void agregar(int cod, int cant, String tit, String aut, float pre)
	{
		nodo aux;
		aux = primero;
		primero = new nodo(cod, cant, tit, aut, pre);
		primero.setProximo(aux);
	}

	// metodo que tomando un código de libro pueda eliminarlo de la lista
	// punto b) del ejercicio uno
	public void eliminar(int cod)
	{
		nodo aux1, aux2, aux3;
		// veamos si el que hay que eliminar es el primero
		if(primero.getCodigo()==cod)
		{
			aux1 = primero.getProximo();
			primero.setProximo(aux1);
		}
		else // no es el primero, ahy que buscar antes de borrar
		{
			aux1 = primero;
			for(aux2 = primero.getProximo();aux2 != null && aux2.getCodigo() != cod;
aux2=aux2.getProximo())
			{
				aux1 = aux2;
			}
			if(aux2 != null) // encontro el nodo buscado!!!
			{
```

```
                        aux3 = aux2.getProximo();
                        aux1.setProximo(aux3);
                }
        }
}

// metodo qeu retorna la cantidad de libros de la lista
// punto c) del ejercicio uno
public int getCantidad()
{
        int cant = 0;
        nodo aux1;
        for(aux1 = primero; aux1 != null; aux1=aux1.getProximo())
        {
                cant ++;
        }
        return cant;
}

// metodo que retorna la cantidad de libros con mas de X paginas
// punto d) ejercicio uno
public int getCantidadMasPaginas(int cant)
{
        int cantidad = 0;
        nodo aux1;
        for(aux1 = primero; aux1 != null; aux1=aux1.getProximo())
        {
                if(aux1.getPaginas() > cant)  // es de los que buscamos....
                  cantidad ++;
        }
        return cantidad;
}

// metodo que retorna el código del libro de mayor cantidad de pagians
// punto e) del ejerciocio uno
public int getCodigoMasPaginas()
{
        int cant, cod;
        nodo aux1;
        // tomamos como valores iniciales los del primer nodo de la lista
        cant = primero.getPaginas();
        cod = primero.getCodigo();
        for(aux1 = primero; aux1 != null; aux1=aux1.getProximo())
        {
                if(aux1.getPaginas() > cant)  // tiene mas paginas que los otros
                  cod = aux1.getCodigo();
        }
        return cod;
}

// metodo que retorna verdadero si un libro esta en la lista
// punto d) del ejercicio uno.
public boolean estaElTitulo(String tit)
{
        boolean esta = false;
        nodo aux1;
```

```
                for(aux1 = primero; aux1 != null; aux1=aux1.getProximo())
                {
                        if(aux1.getTitulo() == tit)  // es de los que buscamos....
                          esta = true;
                }
                return esta;
        }


        //meto toStrin de la lista como para poder ver que datos tiene...
        public String toString()
        {
                String retorno;
                retorno = "";
                nodo aux1;
                for(aux1 = primero; aux1 != null; aux1=aux1.getProximo())
                {
                        retorno += aux1.toString() + "  -  ";
                }
                return retorno;
        }

} // fin de la clase listaLibros



// un pequeño programa como para ver algunos resultados...
class programa
{
 public static void main(String args[])
 {
        int resultado;
        boolean esta;
        listaLibros lista = new listaLibros();
        lista.agregar(10, 150, "Programacion en Java", "Jorge Tymoschuk", 150);
        lista.agregar(20, 90, "Base de datos", "Silvio Serra", 120);
        lista.agregar(25, 130, "Como aprobar AED", "Mandrake", 1000);
        lista.agregar(4, 35, "Estudie corte y confección", "Utilisima", 25);
        System.out.println(lista.toString());
        lista.eliminar(10);
        System.out.println(lista.toString());
        resultado = lista.getCantidadMasPaginas(100);
        System.out.println("La cantidad de libros con más de 100 páginas es: " + resultado);
        resultado = lista.getCodigoMasPaginas();
        System.out.println("El código del libro con más páginas es: " + resultado);
        esta = lista.estaElTitulo("programacion en java");
        if(esta = true)   // el libro si esta
    System.out.println("El libro Programacion en java si esta en la lista");
   else
    System.out.println("El libro Programacion e n java no esta en la lista");
 }
}
```