TRABAJO PRÁCTICO Nº2
Agregación, Composición, Polimorfismo (Listas)

AGREGACIÓN*("tener un")*
Recordatorio: Relación entre clases en donde una clase contiene una o más instancias de otra. ***La clase contenida puede existir independientemente****. Se referencia en UML con una línea con un rombo vacío.*

EJERCICIO 1

Se necesita un sistema para un hospital. Se debe modelar la clase "Habitación", la cuál tiene una cantidad de camas, un atributo qué indica si está completa, y con una lista de pacientes internados. A su vez, de los pacientes, se debe guardar su nombre, apellido y dni. Los pacientes del hospital pueden, o no, necesitar internación.
Crear los métodos **internarPaciente()** y **darDeAltaPaciente()** qué agregue/quite a un paciente a la lista de pacientes de una habitación. Se debe validar si la misma está o no completa.

EJERCICIO 2

Se debe crear un sistema para un taller mecánico. El mismo debe modelar la clase auto, la cual cuenta con patente, marca, modelo (año fabricación), y un atributo de tipo Motor qué tenga número de motor, kilometraje y cilindrada(nafta, aceite, etc).
Al taller mecánico le llegan para ser reparados Autos (con sus respectivos motores) y Motores solos sin carrocería.

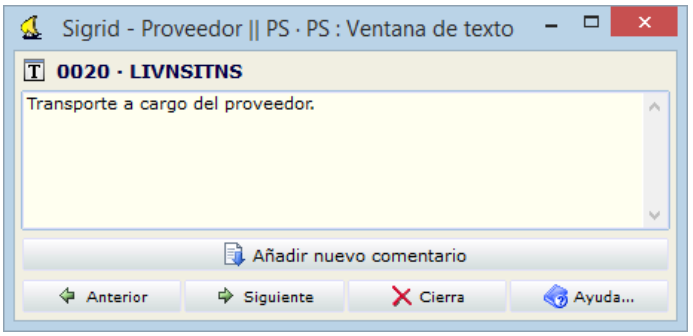
COMPOSICIÓN *("parte de")*
Recordatorio: Relación entre clases en donde una clase contiene una o más instancias de otra. ***La clase contenida no puede existir independientemente de su clase contenedora****. Se referencia en UML con una línea con un rombo relleno. Utilice* ***clases internas*** *(inner classes) para luego declarar los atributos correspondientes.*

EJERCICIO 3

Modelar una clase Casa. La misma está formada por **dormitorios** (int: metros cuadrados,boolean: baño en suit, int: cantidad de ventanas), **baños** (boolean: bañera, boolean: antebaño), **patio**(String: frente o fondo, int: metros cuadrados), **cocina**(tipo horno, electrico o a gas, metros cuadrados, etc).

EJERCICIO 4

Modelar la clase Ventana. Debe contener al menos cinco botones, dos barras de desplazamiento (vertical y horizontal), un marco para mostrar texto. Ver como ejemplo la figura de la derecha.



POLIMORFISMO Y LISTAS

EJERCICIO 5

Escriba las clases necesarias para crear una lista de figuras. De la clase **abstracta** que modela modela las figuras (**Shape**), deben descender clases que modelarán círculos (**Circle**), rectángulos (**Rectangle**) y triángulos (**Triangle**). Use la clase ancestro de las figuras para crear una lista de figuras, para esto debe escribir otra clase (**ShapeList**). La clase que modela la lista de figuras debe tener métodos para:

- Dibujar todas las figuras
- Cambiar el tamaño de todas las figuras (aumentar o disminuir su tamaño) mediante un factor de escala.
- Mover todas las usando dos valores: el "delta" x y el "delta" y.
- Dos métodos que realicen las dos acciones de arriba (escalado y movimiento), pero agregando el número de posición de la figura para que los cambios se realicen solo sobre esa figura elegida.
- Agregar una figura al final de la lista
- Insertar una figura en una posición dada.
- Devolver la cantidad de figuras que contiene.
- Quitar una figura de una posición dada.

En la clase principal debe mostrar que cumple con todas las funcionalidades pedidas. En las llamadas a los métodos puede imprimir texto para mostrar su llamada (Ej.: al llamar al método **dibujar**() de un círculo se mostraría el cartel "dibujando el círculo de radio x…")

EJERCICIO 6

Modelar un sistema para el área de RRHH de una empresa qué guarde información de todos los empleados. Todos tendrán id, nombre y apellido, edad, fecha de ingreso y sueldo. Los mismos pueden ser "Administrativo", "Operario Maestranza" y "Vendedor" y deben extender de la clase abstracta "Empleado". Deberán tener un método qué muestre por pantalla qué tareas realizan.
Crear la clase "BaseEmpleados" qué contenga una lista con todos los empleados de la empresa. Crear los métodos:

- agregarEmpleado(Empleado), para crear y cargar un nuevo empleado a la lista.
- eliminarEmpleado(id), para eliminar un empleado según un id.
- mostrarTodosLosEmpleados(), para listar por consola todos los empleados, mostrando nombre y apellido y fecha de ingreso.
- buscarEmpleado(nombreEmpleado), para Imprimir por consola todos los datos de un empleado dado su nombre.
- verCantidadEmpleados(), qué muestre por pantalla la cantidad de empleados qué tiene la empresa.
- mostrarEmpleadosPorTipo(String), para mostrar por consola solo los empleados de un tipo en particular (o administrativo, o maestranza, o vendedor).

Crear una clase qué contenga el método main() y:

- crear una lista de empleados vacia
- crear 2 empleados de cada tipo con el metodo agregarEmpleado()
- imprimir por pantalla la lista de todos los empleados
- mostrar los empleados del tipo administrativo con el metodo mostrarEmpleadosPorTipo()
- Eliminar 2 empleados
- buscar un empleado por nombre
- ver cantidad de empleados.