***ENUNCIADOS POO***

***EJERCICIO 1***

**Descripción**: Crea una clase abstracta llamada Animal con un método abstracto hacerSonido().

1. Crea clases concretas que hereden de Animal:
   1. Perro: su método hacerSonido() imprime "Guau".
   2. Gato: su método hacerSonido() imprime "Miau".
2. Crea una clase distinguida MainEjercicio4 y en su método main()crea una lista de objetos Animal, agrega varios Perro y Gato, e itera sobre la lista llamando a hacerSonido() para cada uno.

***EJERCICIO 2***

**Descripción:** Repite el ejercicio anterior pero utilizando una interfaz Animal con un método hacerSonido(), en lugar de una clase abstracta.

1. Crea clases que implementen la interfaz Animal:
   1. Perro: su método hacerSonido() imprime "Guau".
   2. Gato: su método hacerSonido() imprime "Miau".
2. Crea una clase distinguida MainEjercicio4 y en su método main()crea una lista de objetos que implementen la interfaz Animal, agrega varios Perro y Gato, e itera sobre la lista llamando a hacerSonido() para cada uno.

***EJERCICIO 3***

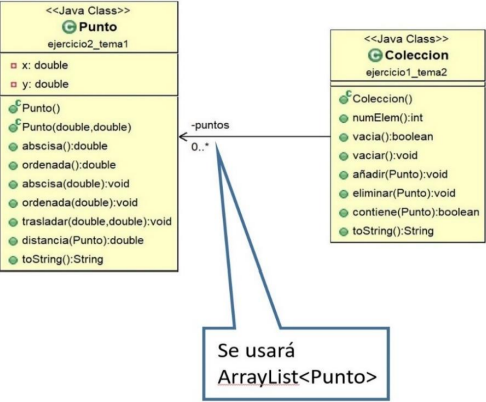
**Descripción:** Crea una jerarquía de clases para representar vehículos:

1. Clase base Vehiculo con los atributos: o Marca (String) o Modelo (String) o Año (int)
2. Clase Coche que herede de Vehiculo y agregue el atributo: o Número de puertas (int)
3. Clase Motocicleta que herede de Vehiculo y agregue el atributo: o Cilindrada (int)
4. Crea una clase distinguida MainEjercicio3 y en su método main()crea al menos un coche y una motocicleta, y muestra su información.

Cada clase debe tener un constructor para inicializar sus atributos y un método mostrarInfo() que imprima los datos del vehículo.

***EJERCICIO 4***

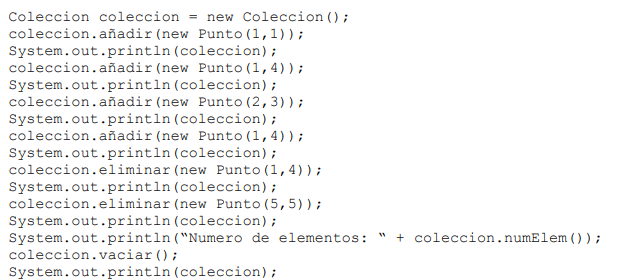
**Descripción:** Dentro del proyecto crea el paquete ejercicio1\_tema2. En este paquete, diseña la clase Coleccion que represente una colección de puntos (objetos de la clase Punto diseñada en el ejercicio 2 de la Primera Relación de Problemas del Tema 1). Esta clase tendrá sólo una variable de instancia denominada puntos (private) para almacenar la colección de puntos mediante el uso de ArrayList. Tendrá un constructor sin argumentos (Coleccion()), que crea una colección vacía de tamaño predefinido por el sistema.

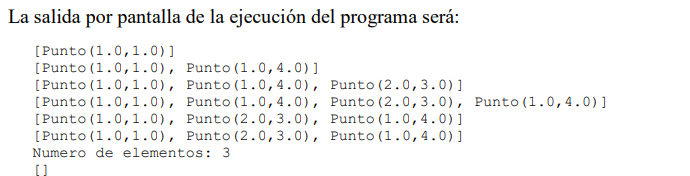


La clase ofrecerá métodos (public) para (además, si se considera necesario, se podrán crear métodos privados):

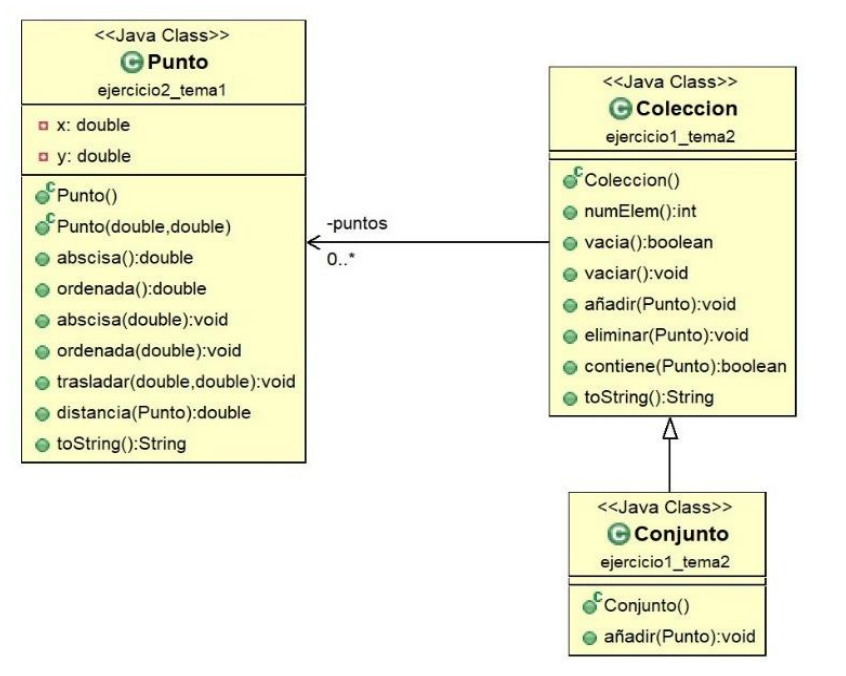
* consultar el número de elementos de la colección (int numElem()).
* comprobar si la colección está vacía (boolean vacia()). - vaciar una colección (void vaciar()).
* añadir un punto a la colección (void añadir(Punto p)).
* eliminar un punto de la colección (void eliminar(Punto p)).
* Si el elemento no está, no se hace nada.
* comprobar si la colección contiene un punto (boolean contiene(Punto p)).
* representar con una cadena de caracteres la colección (String toString()). La representación consistirá en la secuencia de puntos que forman la colección, separados por comas y encerrados entre corchetes.

Para probar el funcionamiento de la clase Coleccion, construye una clase distinguida (dentro del paquete por defecto del proyecto) denominada Prueba con el método main que realice diversas operaciones con una colección de puntos de la siguiente forma:

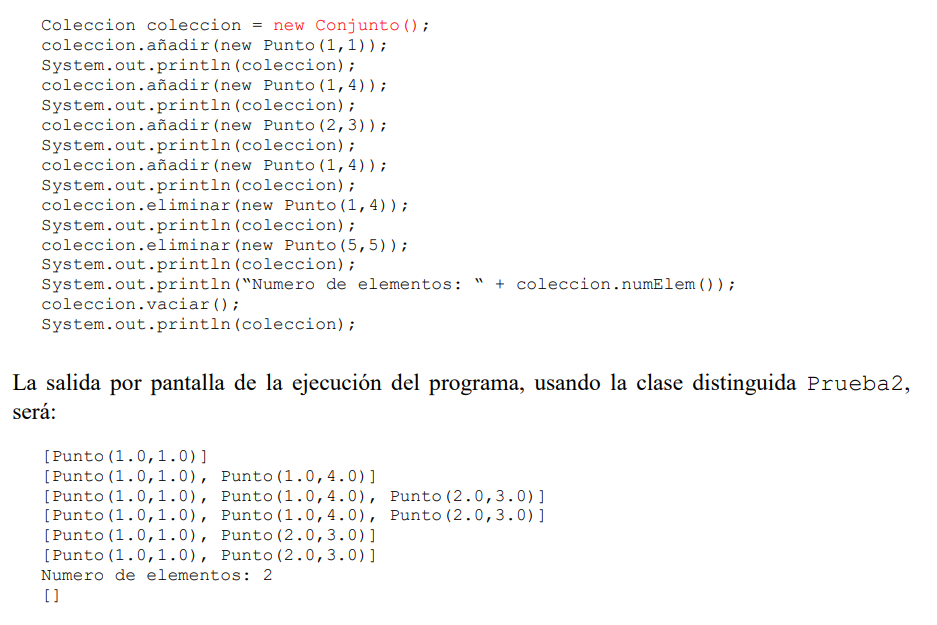




Dentro del paquete ejercicio1\_tema2, diseña otra clase denominada Conjunto que represente un conjunto de puntos. Esta clase debe heredar de la clase Coleccion. No añade ninguna variable de instancia más. Tendrá un constructor sin argumentos (Conjunto()), que crea un conjunto (colección) vacío de tamaño predefinido por el sistema. Además, redefinirá el método añadir de Coleccion para evitar almacenar elementos repetidos (con esto estamos practicando el concepto de “Redefinición del Comportamiento” junto con la “Herencia”).



Para probar el funcionamiento de la clase Conjunto, construye una clase distinguida (dentro del paquete por defecto del proyecto) denominada Prueba2 con el método main que realice diversas operaciones con una colección de puntos de la siguiente forma. Nótese que el código es el mismo de la clase Prueba anterior con la única diferencia de que en la primera línea, se crea un objeto de la clase Conjunto en lugar de un objeto de la clase Coleccion (resaltado en rojo). Con esto, además del concepto de “Herencia”, estamos practicando los conceptos de “Polimorfismo” y “Vinculación Dinámica”.



Nótese la diferencia con la salida anterior (la ejecución de la clase distinguida Prueba). Aquí no se repite el Punto(1.0,4.0).