ENUNCIADO DEL EXAMEN – PRÁCTICAS LTP – 2/11/2021

Resuelve los 4 ejercicios de este enunciado de examen en el proyecto BlueJ facilitado.

IMPORTANTE. Se valorará, cuando sean aplicables, el uso adecuado de los mecanismos de **herencia**, **sobrecarga**, **polimorfismo** o **genericidad**.

Contenido del proyecto BlueJ.

Se facilita el siguiente conjunto de clases e interfaces:

- **Coleccion<T>**: Interfaz que modela las operaciones sobre una colección de elementos.
- **Vector<T>**: Interfaz que extiende a **Coleccion<T>**, añadiendo funcionalidades para acceder arbitrariamente a los elementos de una colección.
- **VectorList<T>**: Clase que implementa la interfaz **Vector**.
- **Figure**. Clase que guarda información sobre figuras geométricas (concretamente, su posición).
- **ColeccionUse**: Clase con un método **main** para probar el resto de clases. Esta clase NO se debe modificar. Esta clase compilará cuando se resuelvan los ejercicios.

EJERCICIO 1 (1,75 puntos).

Crea una nueva interfaz, llamada **Conjunto<T>**, que extienda a **Coleccion<T>** definiendo dos nuevas operaciones:

boolean pertenece(T e)

Conjunto<T> subconjunto(int n)

La funcionalidad de estos métodos:

- **pertenece** es un método que devuelve **true** si **e** es un elemento del conjunto, y **false** en caso contrario.
- **subconjunto** es un método que devuelve un subconjunto de **n** elementos del conjunto.

EJERCICIO 2 (3,5 puntos).

Crea una nueva clase, llamada **ConjuntoList<T>**, que implemente la interfaz **Conjunto<T> c**on las siguientes características:

- Extenderá la clase **VectorList<T>**.
- No dispondrá de atributos (solamente los recibidos por herencia).
- El método **subconjunto** devuelve el subconjunto de los **n** primeros elementos añadidos al conjunto. Si hay menos de **n** elementos en el conjunto, devuelve todo el conjunto.

EJERCICIO 3 (2,75 puntos).

Sobreescribe en la clase **ConjuntoList<T>** el método**:**

void add(T e)

De modo que solamente se añada el elemento **e** al conjunto si dicho elemento **e** no está en el conjunto. Es decir, sobreescribe el método **add** para que en el conjunto no hayan elementos repetidos.

EJERCICIO 4 (2 puntos).

Crea una nueva clase, llamada **FiguresConjunto<T>**, que extienda la clase **ConjuntoList<T>**, pero con restricción de la genericidad a la clase **Figure**. **FiguresConjunto** no dispondrá de atributos (solamente los recibidos por herencia).

TEST DE LOS EJERCICIOS.

Si se resuelven correctamente los ejercicios, la clase **ColeccionUse** compilará y la ejecución de su método **main** generará la siguiente salida:

```
VectorList vL : [8, 7, 6, 5, 4, 3, 2, 1]
ConjuntoList cL : [20.0, 19.0, 18.0, 17.0, 16.0, 15.0, 14.0, 13.0]
ConjuntoList cL : [20.0, 19.0, 18.0, 17.0, 16.0, 15.0, 14.0, 13.0, 99.0]
Pertenecen al ConjuntoList? : (17.0 : true) (99.0 : true) (98.0 : false)
Subconjunto 3 elementos : [20.0, 19.0, 18.0]
Subconjunto 4 elementos : [20.0, 19.0, 18.0, 17.0]
Subconjunto 5 elementos : [20.0, 19.0, 18.0, 17.0, 16.0]
FiguresConjunto cF : [Position: (4.0, 6.5), Position: (3.5, 8.2)]
```

Obtener esta salida es indicio, pero no garantía, de la resolución correcta de los ejercicios.
