Prueba 1 de Evaluación Continua

Análisis y Diseño de Software (2015/2016)

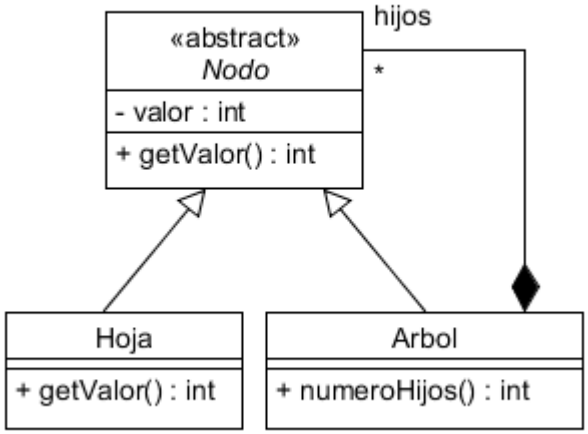
Contesta el ejercicio 1 y 2 en esta hoja, y el 3 en el espacio al dorso

Apellidos:
Nombre:

Ejercicio 1 Herencia y polimorfismo (3 puntos)

Dado el diagrama de clases de la derecha, indica cuál de las siguientes líneas de código Java contiene algún error e indica la razón (debajo de la línea):

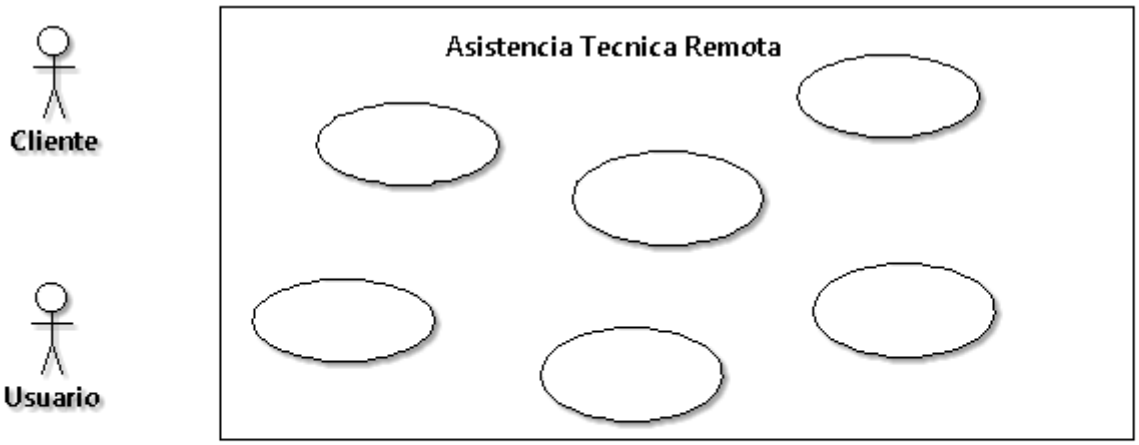
```
1) Nodo hoja = new Nodo();

2) Nodo raiz = new Arbol();

3) Arbol otro = raiz;

4) raiz.getValor();

5) raiz.numeroHijos();

6) raiz.valor = 7;
```



Ejercicio 2: Diagramas de casos de uso (3 puntos)

Considera una aplicación de asistencia técnica y soporte remotos sobre los productos de una empresa. Se permite que cualquier *usuario* consulte los problemas resueltos hasta el momento, realizando búsquedas por el tipo de producto o por las características del fallo. Los *clientes* pueden solicitar ayuda a domicilio. Para ello deberán autentificarse y haber buscado su problema. Los clientes pagarán el servicio a domicilio una vez que éste se ha prestado.

Se pide: completar el siguiente diagrama de casos de uso, para que refleje los requisitos descritos, **indicando el número** correspondiente a cada caso de uso (de los seis dados abajo) y **añadiendo todas las conexiones** necesarias entre actores y casos de uso, y casos de uso entre sí.



1. Buscar por tipo de producto
2. Solicitar ayuda a domicilio
3. Buscar problema resuelto
4. Buscar por características del fallo
5. Pagar el desplazamiento
6. Autentificarse

Ejercicio 3: Diagramas de clase (4 puntos)

Queremos modelar una aplicación para el estudio de ecosistemas, tanto especies como animales concretos. Un ecosistema está formado por varias especies animales. Consideraremos dos subtipos especiales de especie: *presas* y *depredadores*. Las especies están jerarquizadas, por lo tanto (y opcionalmente) una especie puede pertenecer a otra más general. Estas especies generales pueden no ser ni presa ni depredador, por ejemplo "Mamífero", que a su vez pertenece a la especie "Vertebrado". Comprobaciones como que una presa no pueda pertenecer a una especie más general que es depredador, o relaciones entre especies que producirían ciclos, las realizaremos en tiempo de ejecución y no es necesario que las incluyas en el diagrama. Todas las especies se crearán de forma dinámica, indicando su nombre, y a qué especie más general pertenece.

Los depredadores no sólo se alimentan de presas, sino que podrían alimentarse de otros depredadores, o de especies más generales, como "Mamífero". La información sobre qué especies consume cada tipo de depredador se gestionará de forma dinámica. La aplicación deberá por lo tanto permitir asociar depredadores con las especies de las que se alimenta.

Finalmente, queremos poder representar animales concretos de una especie, mediante un identificador, así como datos derivados de su seguimiento en el entorno, como su peso.

Se pide:

(a) Realiza el diagrama de clases que describe la aplicación anterior (3,25 puntos). No es necesario incluir constructores ni getters o setters.

(b) Añade los métodos necesarios para (0,75 puntos):

1. Obtener la lista de especies "hermanas", es decir, que pertenezcan a la misma especie más general.
2. Asociar un depredador con las especies de las que se alimenta.
3. Añadir un nuevo animal concreto al ecosistema, indicando los datos necesarios para su creación.