Práctico 0 - Repaso

1. (Segundo Parcial – Programación 1 – 2016 – Ejercicio 2)

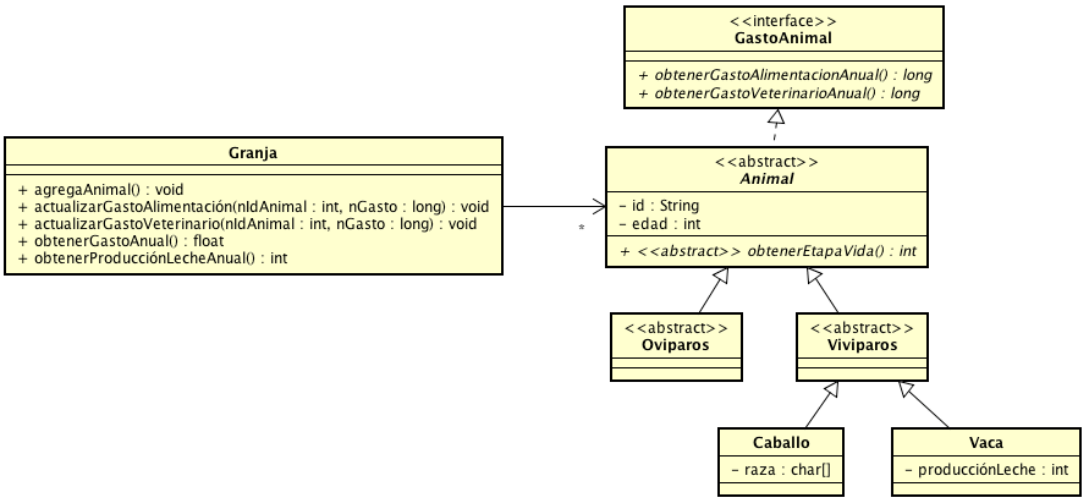
Una granja desea realizar un sistema que le permita tener trazabilidad de sus animales como también determinar los gastos de mantenimiento de la misma. Con respecto a los gastos están divididos en dos grandes áreas, gastos alimenticios y gastos veterinarios, de los cuales se tiene una previsión anual por animal.

A su vez a cada animal se lo clasifica en 3 categorías en base a su edad, estas es CRIA, ADULTO o SENIOR. Para una vaca, se la considera hasta los 2 años como cría, de los 2 años hasta los 15 como adulto y de los 15 en adelante como senior. En el caso de los caballos se lo considera cría hasta los 3 años, de 3 a 18 como adulto, y de 18 en adelante como senior.

La clasificación de edad es de utilidad para determinar el costo veterinario, ya que un animal SENIOR tiene una previsión de un 20% extra dado a su edad y un animal CRIA tiene un 10% extra. Esto quiere decir que si tengo un caballo que tiene una previsión anual de $U 10.000 pero es SENIOR su gasto veterinario será $U 10.000 + $U 10.000 * 20%.

Por último se desea conocer la capacidad de producción de leche, teniendo en cuenta que deberá considerarse sola aquellas vacas que sean adultas.

Teniendo en cuenta la realidad planteada se realizó el siguiente diagrama conceptual en UML:



Listado de operaciones a implementar:

- void agregarAnimal(…) throws AnimalYaExiste;
  - Crea un animal dentro del sistema. Lanza la excepción AnimalYaExiste si ya fue registrado un animal con el id pasado. **Deberá proponer argumentos para la operación que más se adecuen al problema.**
- void actualizarGastoAlimentación(…) throws AnimalNoExiste
  - Actualiza el importe de gasto de alimentación anual para un animal. Considerar que el mismo es un entero.
- void actualizarGastoVeterinario(…) throws AnimalNoExiste
  - Actualiza el importe de gasto veterinario anual para un animal. Considerar que el mismo es un entero.

- float obtenerGastoAnual()
    - Recorre todos los animales y suma el gasto de alimentación y el gasto veterinario. Teniendo en cuenta las consideraciones extras del gasto veterinario indicados anteriormente en la letra.
- int obtenerProducciónLeche()
    - Recorre todos los animales vacunos y suma la producción de leche de cada uno de ellos.

Consideraciones:

- Para todas las clases deberá colocar constructores, get's, set's y operaciones heredadas de Object que considere necesarias.
- La clase caballo es una clase concreta que no deben tener clases hijas.
- La operación obtenerEtapaVida a implementar en Caballo y Vaca no deben permitir ser sobrescritas por clases hijas.
- Las clases Animal, Oviparos, Vivíparos, Caballo y Vaca se deben colocar en el paquete prog1.sparcial.animales. Para la interfaz CostoAnimal y la clase Granja se deben colocar en el paquete prog1.sparcial.
- La operación obtenerEtapaVida dentro del modelo devuelve un entero el cual si es 0 representa una cría, 1 representa un animal adulto y 3 un animal senior.