Anteproyecto de Zoo

Antonio Luque Bravo

1. ***ArrayList:*** *Contendrá un ArrayList de los animales del zoo.*
2. ***Enumeraciones:***
   - *Alimentación: tendrá el tipo de alimentación que tienen los animales del zoo (Carnívoros, Herbívoros y Omnívoros)*
   - *EspeciesMamiferos: contendrá los mamíferos que están en el zoo, solo estará disponible para la clase Mamíferos.(Oso ,Suricato ,Tigre, Marmota)*
   - *EspeciesAves: contendrá las aves que estén en el zoo, solo disponible para la clase Aves. (Avestruz, Águila, Golondrina).*
   - *EspeciesPeces: los peces que puede haber en el zoo,solo disponible para la clase Peces. (Tiburón, Salmón, Carpa, Pez Gato).*
3. ***Herencia:*** *La clase Animal contendrá a sus clases hijas teniendo todos en común un alias, energía,peso, un tipo de alimentación y una fecha. Cuando el animal no tenga suficiente energía, no podrá realizar ninguna acción, tendrá que comer(Interfaz):*
   a. *Mamíferos: Tendrán una enumeración exclusiva de Mamíferos(EspeciesMamiferos), un método que será desplazarse()(Interfaz Desplazable) que gaste energía y peso dependiendo del mamífero, un boolean hibernando que estará true si está hibernando y false si no, con sus correspondientes getters y setters.*
   b. *Aves: Tendrán una enumeración exclusiva de Aves(EspeciesAves), un método que será desplazarse()(Interfaz Desplazable) variará la forma de desplazarse si se trata de un ave que vuela o no, gastará energía y un peso correspondiente al ave y un boolean que indique si el ave puede volar o no.*
   c. *Peces: Implementará una enumeración exclusiva de Peces(EspeciesPeces), un método que será desplazarse()(Interfaz Desplazable) en el cual gaste energía y pierda peso dependiendo del pez que sea.*

- ***Interfaces:***
   a. *Energizable: Aumentará la energía de un animal con el método comer() que aumentará de peso dependiendo de la dieta del animal en cuestión con su energía y el peso que cogerá.*
   b. *Desplazable: Se implementará también en todas las clases hijas y contendrá un método que será desplazarse() que dependiendo de la clase hija a la que se aplique, se desplazará de una forma u otra(mamífero correr, ave volar y correr,*

pez nadar con su consumo de energía y peso que dependerá del animal que se esté desplazando)

- ***Ficheros:*** *Se guardará el almacenamiento de los animales en ficheros.*
- ***Excepciones:***
    a. *CodigoNoValidoException: Cuando se intenta crear un animal con el código erróneo.*
    b. *AnimalSinPesoException: Cuando se intenta crear un animal sin peso.*
    c. *AnimalPesoIncorrectoException: Cuando se intenta almacenar un animal con un peso inválido(es decir, que no sea double).*
    d. *AnimalSinEnergiaException: : Cuando se intenta crear un animal sin energía*
    e. *AnimalEnergiaException: Cuando se intenta almacenar un animal con una energía invalida(es decir, intentar almacenarse si el tipo de dato no es int)*
- ***Expresiones regulares:*** *Alias del animal, el animal tendrá un alias o un nombre el cual comienza con tres letras dependiendo de si es mamífero, ave o pez(MAM para mamífero, AVE para ave y PEZ para pez, siendo tres expresiones regulares para cada una de las clases hijas), seguido de un guión y de una palabra de cinco letras de entre mayúsculas, minúsculas y acentos. Luego un guión y uno o más números de entre 0 y 9.*
- ***Fechas:*** *El día que se almacenó el animal.*
- ***GUI:*** *Tendrá los menús de almacenar animales (Fichero), un menú donde se añada o se elimine un animal (Edición), otro menú donde podamos interactuar con el animal(Interactuar), dentro del menú interactuar se podrá dar de comer al animal aumentando así su peso y energía y poder ver que se ha modificado, hacer que se desplace y muestre su gasto de energía y de peso, listar los animales por raza o por tipo de animal y un menú correspondiente a la ayuda del programa y la información de su creador.*