Mario Castillo

Documentación (Java Species)

- **Nota:** Encima de las clases se podrá notar la palabra importada package, el cual me organiza todas las clases en una sola carpeta el cual es Epacies, para mejor organización (parte del bonus). Y el código esta comentado.
- **Main:** El Main se nos fue dado con la intención de ir descubriendo poco a poco para que funcionaba cada una de las líneas y al final del código todo tiene sentido, ya que al final en el Main es donde se llama todo el código, en donde llama cada una de las clases que tiene por debajo y al final imprime un resultado. Llama a todos los animales por su nombre, además contiene un control de error el cual es que si falta un nombre en algún animal el programa se detiene, luego un dato peculiar de los reptiles es que se llama a la clase para que diga que han soltado sus huevos y también con tiene otro control de error al momento de emparejar los animales con otra especie, que también se controla en la clase gato.
- **Define Class:** En el define es tipo interface, significa, que solo se puede utilizar métodos, el cual no te deja construir objetos dentro de la misma. El cual adentro de ella tengo un método publico que no me devuelve nada ya que es tipo void.
- **Abstract Class:** Esta clase tiene como función principal ya que esta por encima de todas a partir de la clase Animal, ya que representa todos los seres vivos. El cual se extiende por todas las clases y se define en define. Esta clase no define ningún tipo de cuerpo.
- **Animal Class:** La clase animal abarca todas clases que están por debajo de ella, mamíferos y reptiles y luego los animales. Desde las líneas 6-13, son los atributos que uso durante el código, el son tipo String, Int, Static int y Boolean. Primero creo un constructor “Animal” el cual me lanza una Excepcion, el cual indica un mensaje que al Animal le falta un nombre, si al Animal no le falta un nombre, con la variable name llama al animal por su nombre y la variable contador crea el número de instancias en el terminal, el

cual añado en el println. Luego creo un par de Metodos, define, legs, Beborn, que a su vez están explicado con los comentarios en el código, en donde los extiendo desde la clase mas baja hasta esta, para que me pueda imprimir todo de forma ordenada, que es el animal, las piernas, si el animal ha sido creado. Luego el siguiente método es el “setLegs”, el cual es un control de error en donde si el animal tiene patas menor a 0, a excepción de la Snake que, si puede tener patas en 0, da como resultado un error, sino que no diga nada y muestre las patas. El método getLegs funciona básicamente para devolver el método asignado del atributo anterior, es decir, devuelve el resultado patas, que proviene del getLegs. Luego está el siguiente método que es al momento de emparejar al animal, el cual sino está seguro que retorne un False y por último, el “toString” que básicamente es una cadena de caracteres en donde me va añadiendo textos, cada vez que la llame en alguna de las clases siguientes.

- **Mammal Class:** Esta clase contiene solo los animales mamíferos tan como su nombre lo indica, primero creo un constructor de tipo publica, en donde llamo al super, el cual esta clase extiende de Animal que es la clase de arriba. Luego creo los distintos métodos, el define que me indica el mensaje de la especie mamifero, luego las patas, que me indica el mensaje sobre las patas de los mamíferos y el “Suckle” que me indica que si es “True” el animal no esta vivo. Y por último vuelvo a llamar al método toString para añadirle el texto y decirle que viene también desde la clase Animal.
- **Reptile Class:** Similar a la clase Mammal esta clase extiende desde Animal, el cual por debajo de ella solo se encontrarán los Animales Reptiles, primero creo un constructor de tipo publica, en donde llamo al super, el cual esta clase extiende de Animal que es la clase de arriba. Luego con el método define que viene desde el super, imprimo la carateristica principal de los reptiles, luego se crea un método, es que un método especifico de los reptiles el cual se controla con la variable boolena igual que en Mammal, en donde si es igual a “True” me imprima por pantalla que el animal a puesto un huevo.

Y por último llamo al ToString, el cual añado el texto que ha pasado por la clase “Reptile”.

- **Cat Class:** Esta clase Gato extiende desde la Clase de Mamífero. Primero creo un constructor Gato, el cual me lanza una excepción, el cual llamo desde el super (Cat), en donde controlo el Error con Try y cacth, el cual, si el Animal tiene 4 patas, que no pase nada, pero si tiene patas negativas que me imprima el mensaje, que llamo con el “getMessage”. Luego creo los métodos, el cual el primero que es el define, llamo al contador para que me vaya añadiendo los números de las instancias creadas de cada println, luego un println con el nombre del animal que se almacena en el “super.define” y luego un mensaje con la característica del gato. Luego un método de patas, que me indique cuantas patas tiene el gato, el siguiente método es sobre si el gato a trepado un árbol y si es “True” que dé como respuesta que el animal no esta con vida. Luego creo un método con un control de error que es cuando el animal se empareja con otro, el cual lo hago con una variable boolena el cual me crea una instancia en Gato, en donde si el emparejamiento se hizo que me imprima “Pair Ok”, pero luego si se quiere emparejar con otro animal, que me diga el mensaje “Ya ha sido emparejado” y luego un mensaje que dice “Que no se puede emparejar con un animal de una especie diferente”. Luego que retorne falso. Dentro de los métodos están las variables globales que anteriormente definí en Animales. Y por último llamo al toString y añado el texto que ha pasado por la clase Gato.
- **Dolphin Class:** Esta clase Delfín extiende desde la Clase de Mamífero. Primero creo un constructor Delfín, el cual me lanza una excepción, el cual llamo desde el super (Dolphin), en donde controlo el Error con Try y cacth, el cual, si el Animal tiene 4 patas, que no pase nada, pero si tiene patas negativas que me imprima el mensaje, que llamo con el “getMessage”. Luego creo los métodos, el cual el primero que es el define, llamo al contador para que me vaya añadiendo los números de las instancias creadas de cada println, luego un println con el nombre del animal que se almacena en el

“super.define” y luego un mensaje con la característica del Dolphin. Y por último llamo al toString y añado el texto que ha pasado por la clase Dolphin.

- **Snake Class:** Esta clase Snake extiende desde la Clase de Reptile. Primero creo un constructor Snake, el cual me lanza una excepción, el cual llamo desde el super (Snake), en donde controlo el Error con Try y cacth, el cual, si el Animal tiene 0 patas, ya que la serpiente no tiene patas por eso es 0, pero si tiene patas negativas que me imprima el mensaje, que llamo con el “getMessage”. Luego creo los métodos, el cual el primero que es el define, llamo al contador para que me vaya añadiendo los números de las instancias creadas de cada println, luego un println con el nombre del animal que se almacena en el “super.define” y luego un mensaje con la característica del Snake. Luego viene el método “Legs” el cual no me devuelve nada solo un mensaje ya que el animal no está seguro que tiene patas. Luego el siguiente método que solo funciona para los reptiles, el cual es dejar un huevo, en donde este caso la llamo con el super y seguido de ello se podrá ver el mensaje en la terminal. Y por último llamo al toString y añado el texto que ha pasado por la clase Snake.
- **Turtle Class:** Esta clase Turtle extiende desde la Clase de Reptile. Primero creo un constructor Turtle, el cual me lanza una excepción, el cual llamo desde el super (Turtle), en donde controlo el Error con Try y cacth, el cual, si el Animal tiene 4 patas, que no pase nada, pero si tiene patas negativas que me imprima el mensaje, que llamo con el “getMessage”. Luego creo los métodos, el cual el primero que es el define, llamo al contador para que me vaya añadiendo los números de las instancias creadas de cada println, luego un println con el nombre del animal que se almacena en el “super.define” y luego un mensaje con la característica del Turtle. Luego creo un método con variable booleana, el cual, si es verdadera que me imprima lo siguiente “La tortuga esta dentro de su caparazón”, el cual es un rasgo característico del animal. Luego igual que en Snake viene el método “Legs” en donde no me devuelve nada, solo un println que me indica que el animal no sabe si tiene patas. Luego el siguiente método que solo funciona para los reptiles, el cual

es dejar un huevo, en donde este caso la llamo con el super y seguido de ello se podrá ver el mensaje en la terminal. Y por último llamo al toString y añado el texto que ha pasado por la clase Turtle.