**Documentation for the Homework 3**

The project is about OOP concepts, in the last HW, we primarily worked with the inheritance, abstract class, abstract methods along with static data members and static functions. In this homework we’ll revise these concepts along with the interfaces and exception handling too.

**Cleanable.java (interface)**

This is the interface with two methods **wash () & dry ().**

**In this homework we have a Zoo class with the 3 categories of animals in it.**

**Zoo.java**

The Zoo class is a class that represents a zoo. It has a name, a maximum number of cages, a current number of cages, and an array of cages.

1. **Rodent.java**

The Rodent class is an abstract class that extends the Animal class. It has two constructors, one with two parameters and one with one parameter. It has an abstract method chill, which is implemented in the child classes. It has a print method, which is overridden from the Animal class. It has a getNumberOfLegs method, which is overridden from the Animal class. The getNumberOfLegs method returns 4, because all rodents have 4 legs.

It has following child classes:

* 1. GuineaPig.java
  2. Hamster.java

1. **Animal.java**

This class’ code is an abstract class that represents an animal. It implements the Cleanable interface. It has 2 constants, 1 static attribute and 2 static methods. It has 3 attributes, 2 constructors, 5 getters and setters, 1 abstract method, 5 methods and 2 methods from the Cleanable interface.

It has following child classes:

* 1. Hippo.java
  2. Lion.java
  3. Rhino.java
  4. Snake.java
  5. Zebra.java

1. **Cage.java**

The Cage class is a class that represents a cage for animals. It has a name, a maximum number of animals, and a current number of animals. It also has an array of animals. The Cage class implements the Cleanable interface. It has a constructor, getters, setters, and methods to add animals, print the cage, make sounds, eat, sleep, count the total number of legs, clean, wash, and dry. The Cage class is a superclass of the Terrarium class.

It has following child classes:

* 1. GuineaPig.java
  2. Hamster.java

1. **Hw3ProjectMain.java**

We create a class called Animal. This class is abstract, meaning that we cannot create an instance of it. We can only create instances of classes that inherit from it. We create a constructor for the class, which takes a name and a number of legs. We also create a method called makeSound, which is abstract. This means that we must implement it in the classes that inherit from Animal. We also create a method called eat, which is not abstract. This means that we can use it in the classes that inherit from Animal, but we do not have to implement it. We also create a method called sleep, which is not abstract. This means that we can use it in the classes that inherit from Animal, but we do not have to implement it. We also create a method called getName, which is not abstract. This means that we can use it in the classes that inherit from Animal, but we do not have to implement it. We also create a method called getNumberOfLegs, which is not abstract. This means that we can use it in the classes that inherit from Animal, but we do not have to implement it.