1. In a cricket simulation game, cricket players can be a batsman or a bowler. They can have a combination of fielding skills like wicketkeeper, slip, gully, cover (we are not considering all types of fielding). Design and implement Java classes and interfaces for cricket players. Since a player can have multiple fielding skills, make use of Decorator pattern to add these fielding skills to the player. The Decorator base class/interface can have a method addFieldingSkills(). Implement the Factory Method pattern to create these players. A player is created based on the user inputs.

The Player interface should have attributes, list of fielding skills (type:String) and name of player. Player should have the method play() which is different for batsman and bowler. Player should have a method field(). field() method should just list the fielding skills the Player has.

The user is asked to give the number of players. Based on the number, the user is asked to input the type of player (batsman or bowler). Then the user is asked to input the fielding skills separated by comma

Eg. Enter number of players 3
Type of player 1: batsman
Fielding skills for player 1: slip,gully
Type of player 2: bowler
Fielding skills for player 2: gully,cover
Type of player 3: batsman
Fielding skills for player 3: wicketkeeper

2. We are building a library for animals to be used in a game. The animals should have methods to move and cry. The library should support cat, dog and horse. When the library is being used in the game, there should only be maximum of one object/instance for each animal. Implement appropriate design pattern to ensure this.