Farm Application

Matti Mäkelä

Developer test assignment
Design document

Requirements

- ★ On start-up prints out all animals with their attributes
- ★ Print other results for 10 days period
- ★ Animals can friend and unfriend others
- ★ If animal A unfriends B, B unfriends A automatically
- ★ Print “unfriend” events before lunchtime
- ★ Group animals by their favorite food and print them eating it
- ★ Animal A asks B to be friends. B can accept or reject. If B accepts, add to each other's friend list
- ★ Print friend events after lunchtime
- ★ Print table that shows who are friends with each other

- ✓ **Friendship rules:**

1. “New friend” is a random animal from the village yard.
2. “Best friend forever” never unfriended.
3. If an animal has 3 or more friends, the probability to lose friend 90%, the probability to get a new friend 10%
4. If an animal has 2 or fewer friends, the probability to lose friend 10%, the probability to get a new friend 90%
5. Rules 2-4 applied for initiating friend requests and for answering as well

Thoughts and assumptions

- ★ Animals share same attributes name, favorite food and best friend expect Bob the rooster (might bring problems). I can probably use Animal class with these attributes and extend other classes with it. Probably going to add friend list to Animal class too since everyone needs to have it.
- ★ Don’t probably have to worry about the 10-day loop before I’m finished with other tasks. Then just loop ready program 10 times.

- ★ Haven’t coded in Java for a while. Need to find out what could be good ways to store animals and their friends. Animals might have to be in somehow sortable collection because of the grouping for food task. Friend list doesn’t probably have to be in any order, I can just check if value is in the list.
- ★ When getting random animal from the village yard, I’ll have to take into account that it can’t be the same animal that’s making the request and can't add anyone that’s already on the list.
- ★ When removing friend that is chosen by random, have to pick a new target if the random happens to be best friend.
- ★ The values <= 2 and >= 3 contain all the possible friend amounts. Probabilities are always either 90% or 10%.
- ★ The way I interpret rules 3-5 is that I can create a boolean value that decides if friend request is sent and is true 10% or 90% of the times depending how many friends' sender has. I have to create another boolean value that is not connected to previous one, that is true 10% or 90% of the time depending how many friends' receiver has. They are different values and both have to be true in order for them to become friends.
- ★ Removing friends only requires one boolean because the other always has to accept it

Development plan steps

1. Create classes for animals
2. Create animals and add all animals into collection
3. Loop through animals and print them
4. Create some kind of friend list for the animals that keeps track who is friends with who, add best friend forever there
5. For all the favorite foods, find animals eating it and group them by that
6. Design of the friendship table, first column & row have animal names.
7. Friend requests, if accepted make them friends with each other
8. Add X to table if they are on each other's friend lists
9. Animal needs to send unfriend request, that removes them both from each other’s list
10. Create loop that loops 10 times