

GridWorld

Chicken Coop 2

Objective: To add a fox to the chicken and egg simulation in GridWorld.

Background: Foxes are predators of Chickens and they must eat them to stay alive. You are going to use GridWorld to add a fox critter into your chicken and egg cycle from Chicken Coop 1.

Assignment:

You will create a new critter called a **Fox**. Foxes are ageless and do not die of old age like Chickens. Instead, foxes die of starvation when they have not eaten for some time. This is sad because foxes do not reproduce. Only the Omnipotent Being (you) can create a fox. Once a fox is dead, it is gone for good.

Foxes eat chickens and nothing else (not flowers, bugs, eggs, etc). This keeps the chicken population in check, a “circle of life” in the barnyard. A fox always runs after the closest chicken it can find.

The fox must eat chickens on a regular basis or the fox dies of starvation. When a fox has eaten, it lies down and takes a nap. During a nap, it does not move or eat anything. After nap time, the fox’s stomach is empty and it resumes its predatory practice.

1. Create a **Fox** class that extends the **Critter** class. Set its color to be **null**. The **Fox** always moves to an open neighboring location at random and faces the direction it has moved. If it cannot move, then it turns in a random direction.
2. Make the **Fox** run after the closest **Chicken**. Find the closest **Chicken** by using the distance formula $\sqrt{(x_2 - x_1)^2 + (y_2 - y_1)^2}$ then move to the next empty location in that direction. If there are no **Chickens** on the grid, then make a move like 1 above. If several **Chickens** are equidistant, then pick one of those directions at random. If there is no open location in the direction of the closest **Chicken**, then move to an open neighboring location at random.
3. Make the **Fox** eat **Chickens**. The **Chicken** must be in a neighboring location to the **Fox**. If more than one **Chicken** is a neighbor, then the **Fox** picks one **Chicken** at random to eat. The **Fox** “eats” by replacing the **Chicken** with a **Tombstone**. After the **Chicken** is eaten, the **Fox**’s stomach is full and it takes a “nap” in which it does not move or eat for 10 steps. After the nap, the **Fox** is hungry again and chases after **Chickens** like in 2 above.
4. Make the **Fox** die. A **Fox** dies when it has been hungry for too long. A **new Fox** always starts as though it just finished napping and is hungry. Keep track of how many steps the **Fox** takes while hungry (after nap). After 20 steps of being hungry, the **Fox** has now starved to death and is replaced in the same location with a **Tombstone**.
5. Once you have everything working, see if you can create a simulation (world) in which the number of Foxes and Chickens near an equilibrium. Equilibrium happens when the **Chicken** population and the **Fox** population remains constant over a period of time (Steps). I will give a “prize” to whomever I feel creates the best equilibrium.

