Project 4 Critters Lab READ ME
Devin Amatya
dga383
16455
Vidhu Appalaraju
vsa267
16465

Read Me

Main is the controller. The main scans the text file and performs specific commands based on the input. If the input is not any of the specified commands, then the input is an invalid command.

Critter is an abstract class. It is used by the controller and contains some abstract methods. These abstract methods are doTimeStep, fight, toString, and runStats. These are used by the Critter subclasses (Critter1, Critter2, Critter3 and Critter4).

The inherited doTimeStep method determines what a Critter subclass object does during a time step, which ranges from attempting to walk, run, walk or run multiple times, or reproduce.
The inherited fight method determines what a Critter does when it encounters another Critter, and will return a Boolean depending on if it wants to actually fight or not.
The inherited ToString returns a string that represents the Critter subclass being used.
The inherited runStats method displays the current stats of the Critter calling the method.

In the Critter abstract class, the walk, run, reproduce, worldTimeStep, clearWorld, updateWorld, and displayWorld methods are implemented. The data structure we used to hold the Critters was a set, and their positions in the world were placed in a 2D String Array.

In the walk and run methods, we move the Critter one(walk) or two(run) steps in the specified direction if they have not previously moved. They are then placed into a set called Moved, which is used to check if any Critter calling the method has already moved or not. If the Critter is calling these methods from their fight method, they also check to make sure that the Critter does not enter the same space as another.

In the reproduce method, the parent object makes an offspring prior to calling this method. If the parent has sufficient energy to reproduce, it gives the offspring half of its current energy rounded down, cuts its own energy in half rounded up, and then moves the offspring 1 space away from it in the direction specified. The offspring then gets placed into a set called babies, and will get put into the world at the end of the worldTimeStep.

In our worldTimeStep we make all Critters perform their doTimeStep function, settle any encounters between Critters, kill off any Critters without any energy remaining, and place any children made during the time step into the world.
The encounters are dealt with by calling both Critters fight methods, performing whatever methods are within them, and making them fight if they are still in the same position. Those that do not want to fight roll a 0, those that want to fight roll a number between 0 and their current energy, and the winner kills the opposing Critter and takes half its energy.
The children are placed into the world regardless if there is another Critter occupying the space, they will only deal with encounters if they occupy the same space as a Critter after every Critter

has moved in the next time step.

In the clearWorld method, we update the worldMatrix 2D String array to contain only a blank space for every single element, and we empty the entire population set.

In the displayWorld method, we show the current state of the world and all Critters within it by printing out a 2D String array, worldMatrix.