Project 4 – Critters

Main – contains main() which prompts for commands in Critter from the console

- quit – terminates program
- show – displays critter world indicating each critter's coordinates
- step [*count*] – simulates a world time step or *count* time steps
- seed (*long*) – seeds with *long* the random number generator used for random events in Critter
- make (*subclass*) [*count*] – creates a Critter of *subclass* or *count* Critter of *subclass*
- stats (*subclass*) – shows statistics for all Critter of *subclass* in the world

Critter – abstract class that contains the world of critters and their interactions

- Critter1 – New Critter subclass that overrides toString, doTimeStep, and fight
- Critter2 – New Critter subclass that overrides toString, doTimeStep, and fight
- move(*speed*, *direction*) – new Critter method that implements run() and walk() given a *speed*, the number of tiles to move away in *direction*
- resolveEncounter(*list*) – new Critter method that resolves encounter between the first two critters in the *list*

All Critter instances created by the user are stored in the ArrayList *population*. During a word time step, offspring that result from reproduction are stored in the ArrayList *babies* and enter *population* by the beginning of the next time step. Another ArrayList *hasMoved* mirrors *population* and keeps track of whether critters have moved within a world time step. The ArrayList *map* keeps track of where critters are on the coordinate grid by keeping an ArrayList at each coordinate point of all critters that occupy that point and is updated after every critter timestep and encounter.