CS108L Computer Science for All
Module 5 NetLogo Code Cheat Sheet

Command / Variable	Description
reset-ticks	Resets the tick counter. Put in the Setup Procedure *after* **clear-all**
tick	Increments the tick counter for the program (for use in update view). Put in the go procedure
globals [*var1 var2 ...*]	A keyword that can only be used at the beginning of a program, before any functions or procedures. It defines new global variables. Global variables are "global" because they are accessible by all agents and can be used anywhere in the program. Most often, globals is used to define variables that need to be used in many parts of the program. **Example:** **globals [NumTurtles ColorTurtles]** ;; declares two global variables ;; NumTurtles and ColorTurtles.
breed [*plural_name singular_name*]	Defines a breed. It is used at the beginning of the Code tab, before any procedure definitions. The first input in the square bracket defines the name of the agentset associated with the breed (the group of all members of that breed) and is given the plural name of the breed. The second input defines the name of a single member of the breed. Note: Every breed is still a turtle, but turtles can only be one breed. **Example:** **breed [frogs frog]** ;; defines the breed frogs
<*breeds*>-own [*var1 var2 …*] **turtles-own [*var1 var2 …*]** **patches-own [*var1 var2 …*]**	The **<*breeds*>-own** keyword defines the variables belonging to that breed and can only be used at the beginning of a program, before any procedures. If you specify **turtles-own** variables then *all* breeds of turtles will have that variable. Note: Each agent has its own copy of the variable. If turtle 1 has a lifespan of 5, turtle 2 could have a lifespan of 8. Note: You must give the variable an initial value (i.e. in setup) before it can be used. **Example:** **frogs-own [energy]** ;; creates the variable energy exclusively for the breed frogs **turtles-own [lifespan]** ;; creates the variable lifespan for *all* turtles, of all breeds **patches-own [lifespan]** ;; creates the variable lifespan for *all* turtles, of all breeds

count *agentset*	Counts the number of agents in a given agent set. **Example:** **count turtles** ;; counts the number of turtles **show count turtles with [color = red]** ;; shows the number of red turtles in the ;; Command Center
set shape ***"string"***	Used in **ask turtles** or **ask <breeds>** command brackets to set the shape of the agent. **Example:** **ask turtles [set shape "wolf"]** ;; sets all turtles' shapes to wolf
ask <breed> # [*commands*] **ask <breeds> [*commands*]**	Tells the given agent or group of agents (agentset) to run the given commands. **Example:** **ask frog 4 [set color red]** ;; sets frog 4's color to red **ask frogs [right 90]** ;; all frogs turn right 90 degrees
hatch # **hatch # [*commands*]** **hatch-<*breeds*> #** **hatch-<*breeds*> # [*commands*]**	Creates # new turtles or <*breeds*> that are exact copies of its parent. For example, new turtles will have the same color and heading as their parent, and the same agent variable values (e.g. lifespan or energy from above). Only a turtle can **hatch**. The turtles immediately run *commands*. This is useful for giving the new turtles different colors, headings, or whatever. If the **hatch-<*breeds*>** form is used, the new turtles are created as members of the given breed. Otherwise, the new turtles are the same breed as their parent. **Example:** **ask turtles** **[** **hatch 1** ;; every turtle creates one new turtle, a copy of itself **[** **left 45** ;; the new turtles turn left and move away **forward 1** **]** **hatch-sheep 1 [set color black]** ;; every turtle then creates a single black sheep **]**