|  |  |
| --- | --- |
| **CS108L**  **Module 6: NetLogo Command Cheat Sheet** | |
| **Command** | **Command 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** [*variable\_names*] | 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 or constants that need to be used in many parts of the program.  Example: **globals [NumTurtles ColorTurtles]** ;; declares two global variables NumTurtles and ColorTurtles. |
| **turtles-own** [*variable\_names*] | A keyword that can only be used at the beginning of a program, before any functions or procedures. It defines new turtle variables unique to each turtle  Example: **turtles-own [eyes legs]** ;; declares two turtle variables eyes and legs. |
| **patches-own** [*variable\_names*] | A keyword that can only be used at the beginning of a program, before any functions or procedures. It defines new patch variables unique to each patch  Example: **patches-own [patchColor]** ;; declares one patch variable patchColor. |
| **links-own** [*variable\_namse*] | A keyword that can only be used at the beginning of a program, before any functions or procedures. It defines new link variables unique to each link  Example: **links-own [traffic];;** declares one link variable traffic. |
| **count** *agentset* | Counts the number of agents in a given agent set  Example**: count turtles** ;; counts the number of turtles  **count turtles with [color = red]** ;; counts the number of red turtles |

|  |  |
| --- | --- |
| **let** *local\_variable\_name* value | Creates a new local variable called *local\_variable\_name* and gives it the given value. A local variable is one that exists only within the enclosing block of commands such as a procedure or within the ask turtles brackets.  **Example: let num1 10**  This creates a local variable num1 and gives it an initial value of 10 |
| **set** *variable\_name* value | Sets variable (*variable\_name*) to the given value.  **Example : set num1 25**  Changes the value of the local variable num1 to 25 |

|  |  |
| --- | --- |
| **breed [ plural\_name singular\_name ]** | This keyword 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 member of that breed)– it is given the plural name of the breed. The second input defines the name of a single member of the breed.  Example: **breed [frogs frog]** ;; defines the breed frogs, with the individual member of the breed being called frog |
| **create-*<breeds>* *number***  **create-*<breeds>* *number* [ *commands* ]** | Creates *number* new breed\_agents at the origin. New breed\_agents have random integer headings and the color is randomly selected from the 14 primary colors. If *commands* are supplied, the new breed\_agents immediately run them. This is useful for giving the agents a different color, heading, etc. (The new agents are created all at once then run one at a time, in random order.)  Example: **create-snakes 10** ;; creates 10 snakes  **create-frogs 15 [set color green]** ;; creates 15 green frogs |
| **set-default-shape turtles “*string*”**  **set-default-shape <*breeds>* “*string”*** | Specifies a default initial shape for all turtles, or for a particular breed of turtles. When a turtle or a breed is created, or it changes breeds, it shape is set to the given shape. This command doesn't affect existing agents, only agents you create afterwards. The given breed must be already defined breed. The given string must be the name of a currently defined shape, which can be created in the Turtle Shapes Editor or chosen from the Turtles Shapes Editor library. In new models, the default shape for all turtles is "default". Note that specifying a default shape does not prevent you from changing an agent's shape later. Agents don't have to be stuck with their breed's default shape.  Example: **set-default-shape cars “car” ;;**sets the default shape of the breed cars to car shape available in the Turtle Shape Editor Library |
| **set shape “ShapeName”** | Used in the ask turtles or ask <breeds> command brackets to set the shape of a turtle or a breed.  Example: **ask turtles [ set shape "wolf" ]** ;;sets turtle shape to wolf |
| **ask <breed> # [set *attribute* #]**  **ask <breeds> [set *attribute* #]** | Tells the given agent or group of agents (agentset) to set a specific attribute (NEtLogo specified agent variable) to a particular value  Example: **ask frog 4 [set color red]** ;; sets frog 4’s color to red  **ask frogs [set size 5]** |

|  |  |
| --- | --- |
| **ask <*breeds>* [*commands…*]**  **ask <*breed> #* [*commands*…]**  **ask <*breeds>* with [**  ***condition*][*commands*]** | Tells the given agent or group of agents (agentset) to set a run the given commands  Examples**: ask frogs [forward 10]** ;; asks all the frogs to move forward 10 steps  **ask frog 2 [right 90]** ;; has frog 2 turn right 90 degrees  **ask frogs with [color = red] [hatch 1]** ;; tells all RED frogs to hatch 1 frog |
| ***<breeds>*-own [*var1* ...]** | The *<breeds>*-own keyword defines the variables belonging to each that breed and can only be used at the beginning of a program, before any procedures. It you specify turtles-own variables then all breeds of turtles will have that variable.  Example: **frogs-own [energy]** ;; defines a variable energy for the breed frogs |
| **hatch #**  **hatch # [commands]**  **hatch-<*breeds>* #**  **hatch-*<breeds>* # [commands]** | The **hatch #** is used to create *number* new turtles from EXISITING turtles. Each new turtle inherits of all its variables, including its location, from its parent. (Exceptions: each new turtle will have a new who number (agent ID number), and it may be of a different breed than its parent if the hatch-*<breeds>* form is used.) 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  The new turtles then run *commands*. You can use the commands to give the new turtles different colors, headings, locations, etc. (The new turtles are created all at once, then run one at a time, in random order.)  Examples: **hatch 1 [ left 45 forward 1 ]** ;; creates one new turtle and the child turns and moves away  **hatch-sheep 1 [ set color black ]** ;;this turtle creates a new turtle of the sheep breed and sets its color to black |
| **sprout #**  **sprout # [commands]**  **sprout-*<breeds>* #**  **sprout-*<breeds>* # [commands]** | Creates *number* new turtles on a patch. The new turtles have random integer headings and the color is randomly selected from the 14 primary colors. The turtles immediately run *commands*. This is useful for giving the new turtles different colors, headings, or whatever. (The new turtles are created all at once then run one at a time, in random order.)  If the sprout-*<breeds>* form is used, the new turtles are created as members of the given breed.  Examples:  **sprout 5** ;; sprouts 5 turtles  **sprout-wolves 10** ;; sprouts 10 wolves  **sprout 1 [ set color red ]** ;; sprouts 1 turtle colored red  **sprout-sheep 1 [ set color black ]** ;; sprouts one black sheep |