**LOGO (LOGARITHM OF GRAPHICS ORIENTED)**

## Commands

| **Command** | **Abbreviations** | **Syntax** | **Output** | **Example** |
| --- | --- | --- | --- | --- |
| SHOWTURTLE | ST | show turtle | It shows the turtle after it is hidden from the screen. | "st" |
| RIGHT | rt | RIGHT(space) <degrees to rotate towards right | Used to turn the turtle right for no. of degrees specified | "right 228" or "rt 228" |
| PENUP | PU | (set) pen up | It sets the turtle to move without drawing. | "pu" |
| PENDOWN | PD | (resets) pen | Resets to a drawing pen when ordered to move. | "pd" |
| LEFT | lt | LEFT(space) <degrees to rotate towards left > | Used to turn the turtle right for no. of degrees specified | "left 228" or "lt 228" |
| HOME | N/A | home | It just comes to the center of the screen but does not cleans the screen | "home" |
| HIDETURTLE | HT | hide turtle | It hides the turtle and helps to view a clear drawing on the screen. | "ht" |
| FORWARD | fd | FORWARD(space)<no. of steps to move forward> | Used to move the turtle front for no. of times specified | "forward 100" or "fd 100" |
| CLEARTEXT | CT | clear text | It clears all the text in the command screen. | "ct" |
| CLEARSCREEN | CS | CLEARSCREEN | It cleans the screen of trails and comes to the center of the screen | "cs" |
| CLEAN | N/A | clean | It cleans the screen of trails but the turtle does not come to the center of the screen | "clean" |
| BACK | bk | BACK(space) <no.of steps to move backwards> | Used to move the turtle back for no. of times specified | "back 100" or "bk 100" |

**PU**

Short for "pen up," it lifts the "pen" from the screen so that moving the turtle doesn't draw a line. *Example:***PU**

**PD**

Puts the pen down so that moving the turtle draws a line. *Example:***PD**

**SetPenSize [n n]**

Sets the width of the pen to **n** pixels. Note that it is necessary to put in two numbers. *Example:***SetPenSize [5 5]**

**SetPC [r g b]**

Sets the pen color to the appropriate RGB (Red, Green, Blue) values, where **r**,**g**,and **b** are numbers that range from 0 to 255. (Non-integers are rounded.)

*Example:***SetPC [255 0 0]***(Gives red)*

**Penerase**

Sets the pen to down and sets the mode to erase. When the pen is moved, it will erase whatever is under it. An abbreviation is **pe.**

*Example:***Penerase** *or*  **pe**

**Pennormal**

Sets the pen back to normal mode and cancels erase mode. *Example:***Penerase**

**setfloodcolor** **[r g b]**

Sets the flood color  to the appropriate RGB (Red, Green, Blue) values, where **r**,**g**,and **b** are numbers that range from 0 to 255. (Non-integers are rounded.)

*Example:***setfloodcolor** **[255 0 255]***(Gives magenta)*

**fill**

Floods the area bounded by lines with whatever color was specified in the setfloodcolor command. *Example:***fill**

http://derrel.net/ep/images/blueline.gif  
**Move (drawing) commands**  
http://derrel.net/ep/images/blueline.gif

**FD** **x:** Move forward **x** pixels. *Example:***FD** **100**

**BK** **x**

Move Backward **x** pixels. *Example:***BK** **100**

**LT** **x**

Rotate the turtle x degrees left. Example: LT 45

**RT** **x:** Rotate the turtle **x**degrees right. *Example:***RT** **45**

**ARC** **a r**

Draw an arc with an included angle of **a** degrees and radius of **r**. However, the turtle remains at the center of the arc.

*Example:***ARC** **45 100**

**ARC2** **a r**

Draw an arc with an included angle of **a** degrees and radius of **r**. However, the turtle ends up at the end of the arc.

*Example:***ARC** **45 100**

**http://derrel.net/ep/images/blueline.gif****Turtle and Position Commands  
http://derrel.net/ep/images/blueline.gif**

**ST:** Shows the current turtle. *Example:***ST**

**HT:** Hides the current turtle. *Example:***HT**

**Orientation**

Returns a three member list with the orientation of the turtle. In two dimensions, we are only concerned with the last element in the list. You can either assign the list to a variable, or get a value from the list.

*Example:***Orientation**;

**Make "startangle LAST Orientation**(Assigns the value of the turtle's x position to the variable *startangle*)

**Pos**

Returns a two member list with the x and y position of the turtle. You can either assign the list to a variable, or get a value from the list.

*Example:***POS**;

**Make "xstartposition FIRST POS**(Assigns the value of the turtle's x position to the variable x*startposition*)

**setorientation [*roll pitch heading*]**

Uses a three element list to set the position of the turtle. In two dimensions, we are only concerned with the ***heading*** element in the list but all three elements are needed.

*Example:***setorientation [0 0 90]**Leaves the turtle pointing 90 degrees from straight up;

**setpos [*x y*]**

Sets the absolute ***x***and   position of the turtle. If the pen is down, it will draw a line from it's previous position.

*Example:***setpos [*100 90*]** Sets the turtle x=100 and y=90.

**SetTurtle n**

Changes to turtle **n** with the first turtle being turtle 0 and the last being turtle 1023. Note that it creates all of the turtles between 0 and the one specified, so **SetTurtle 100**will create turtles 1-99 if they have not yet been created.

*Example:***SetTurtle n**

**SetTurtle n**

Changes to turtle **n** with the first turtle being turtle 0 and the last being turtle 1023. Note that it creates all of the turtles between 0 and the one specified, so **SetTurtle 100**will create turtles 1-99 if they have not yet been created.

*Example:***SetTurtle n**

**http://derrel.net/ep/images/blueline.gif****Program Commands  
http://derrel.net/ep/images/blueline.gif**

**To** **name *arg1 arg2 ....***

Starts a procedure, called **name,** that takes optional arguments ***arg1 arg2 ....***.

*Examples:***To** **Square**

**To** **rpolygon :numsides :sidelength :numrepeats**

**End:** Marks the end of a procedure and is required. *Example:***End**

**CS:** An abbreviation for clear screen, it clears the screen and returns the mouse to it's home position. *Example:***End**

**Repeat n [ instruction list]**

Repeats the actions listed in the **instruction list**an **n** number of times.

*Examples:***Repeat n [ fd 10 lt 90 square ]***or*

**Repeat :numrepeat [ fd 10 polygon :angle :sidelength ]**

**Show:** Shows in the commander whatever you ask. It can be a value or an instruction list.

**Wait n**

Pauses for the specified amount of time, measured in 1/60 seconds, before executing the next command. So, if **n** is 60, the program will pause for 1 second.

*Example:***Random 30**

**http://derrel.net/ep/images/blueline.gif****Mathematics and Variables Commands  
http://derrel.net/ep/images/blueline.gif**

**Random n**

Returns an integer random number that ranges from 0 to just less than**n**. For example, if **n** is 10, the random numbers will be 0, 1, 2, 3, 4, 5, 6, 7, 8, or 9.

*Example:***Random n**

**Make "*name* x**

Assigns the value **x** to the string ***name***. Notice the double quotes in front of ***name***? Those are necessary to tell Logo that ***name*** is a string. However once you have defined ***name***as a variable you may reference that variable by using the colon before ***name***, i.e. ***:name***

*Examples:***Make "side 15.5**

**Make "halfside :side/2**

**http://derrel.net/ep/images/blueline.gif  
Program Flow Commands**

**http://derrel.net/ep/images/blueline.gif**

**FOR [*i start stop step*][*Instruction List*]**

This is the classic For-Next loop. The first word, ***i***, is a variable the holds the current value of the counter. Traditionally, it is ***i*** but it can be any variable. You can get the value of the counter just by referring to the variable. The other three words tell where to start, where to stop, and what the step should be. The step is optional and will be set to 1 or -1 if omitted. The instruction list is the list of instructions that will be repeated whenever the loop executes.

*Examples:***FOR [i 0 :angle ][square rt 1 wait 1]** *This will execute the procedure***square***and turn right one degree each time it through the loop. The loop stops once the value of the variable****:angle****is reached.***FOR [counter 0 15 4 ][show :counter]***Try it and see.*

Other commands that help you control program flow are: **do.while**, **If**,**Ifelse**Look these up in the help file for the program.