|  |
| --- |
| **Python Syntax Reference:** |
| |  |  |  | | --- | --- | --- | | **Statement** | **Example** | **Behavior** | | import module\_name as name\_space | **import ti\_rover as rv** | Required for all TI Rover Python programs. Imports the ti\_rover module into the Python program. The module provides the methods for controlling the Rover.  Sets the current position of the RV as the origin and the heading as 0 degrees measured from the x-axis. The import ti\_rover as rv statement is available from the Fns>Modul>ti\_rover>Drive menu. | | from module\_name import \* | **from ti\_system import \*** | Imports all the functions in the **ti\_system** module for use in the program. It is necessary to import the ti\_system module to use disp\_at( ), sleep( ) and **while not escape( ):**.  The from ti\_system import \* statement is available from the Fns>Modul>ti\_rover>Commands menu. | | rv.forward(distance) | **rv.forward(10)** | Rover drives 10 units forward. The default unit is 10 cm. | | rv.backward(distance) | **rv.backward(10)** | Rover drives 10 units backward. The default unit is 10 cm. | | rv.stop() | **rv.stop()** | Rover stops. This function is executed as soon as Rover receives it. | | rv.color\_rgb(red,green,blue) | **rv.color\_rgb(255,0,0)** | Turns the color LED on with the color red. Values for the red, green and blue LED components are between 0 (off) and 255 (full power). The rv.color\_rgb() function is available from the Fns>Modul>ti\_rover>I/O>Outputs menu. | | sleep(seconds) | **sleep(3)** | The calculator will wait 3 seconds before moving to the next line in the program. sleep() is available from the Fns>Modul>ti\_rover>Commands menu. | | var=rv.waypoint\_x() | **x=rv.waypoint\_x()** | Stores the current x-coordinate position in units of the TI-Rover into the variable *x*. The default unit is 10 cm. rv.waypoint\_x() is available from the Fns>Modul>ti\_rover>I/O>Path menu. | | **Statement** | **Example** | **Behavior** | | disp\_at(row,”text”,”align”) | **disp\_at(3,“x position = "+str(x),"left")** | The **disp\_at()** function displays a text string on a specified row with an alignment of left, center or right. When variable **x** has a value of 7.6, the following is displayed on row 3, aligned to the left: x position = 7.6  disp\_at( ) is available from the Fns>Modul>ti\_rover>Commands menu.  Note: The **str()** function converts a numeric value to a string. The **+** operator is used to join two strings.  **str()** is available from the Fns>Type menu. | | **while not escape( ):**  block | **while not escape( ):**  **x=rv.waypoint\_x()**  **disp\_at(3,"x position ="+str(x),"left")** | Defines a while loop that will continue until the [clear] key is pressed.  While loops repeat the statements in the block if the condition at the top of the loop is true. In the example, looping continues until the [clear] key is pressed to escape the loop. Not pressing a key or pressing any key but escape means that “**not escape( )**” will return “true”. True for the loop condition means that looping continues. If [clear] is pressed “**not escape( )**” will return “false”. False for the loop condition means that looping stops. Program execution skips to the statement just after the loop. Note: The block starts with a **colon** and includes the indented lines that follow. **while not escape( ):** is available from the Fns>Modul>ti\_rover>Commands menu. | | <Boolean expression>  value 1 operator value 2 | **2+3==6 (result is false)**  **x+4>=y (if x=1 and y=3, the result is true)**  **“enter”!=”esc” (result is true)** | Boolean expressions evaluate to either true or false.  Note: == is the Python operator to check equality. >= is the Python operator to check whether the value to the left is greater than or equal to the value on the right. != is the Python operator to check inequality. Boolean operators are available from the Fns>Ops menu or from the [2nd] [test] menu on the keyboard. | | if <Boolean expression>:  block | **if 0<x<2:**  **rv.color.rgb(255,0,0)** | Checks to determine if the value of variable **x** is between 0 and 2. If the statement is “true” then the statements in the **if** block are executed. Otherwise, the block is skipped. In the example, when the value for the variable x is between 0 and 2, the calculator will send a command to the TI-Innovator to set the color rgb LED to be red. | | if <Boolean expression> and <Boolean expression>:  block | **If x>=2 and x<4:**  **rv.color.rgb(0,255,0)** | If both expressions are true the **and** function is “true”, then the block is executed. Otherwise, the **and** function returns false, and the block is skipped. In the example, when the value for x is greater than or equal to 2 and less than 4, the calculator will send a command to the TI- Innovator to set the color rgb LED to be green. **and**, **or** and **not** are available from the Fns>Ops menu or from the [2nd] [test] menu on the keyboard. | |

See the Rover module section beginning on page 26 of the [Python Programming for the TI-84 Plus CE Python Graphing Calculator Guidebook](https://education.ti.com/download/en/ed-tech/1424CF4F539A4DBB9145E2AA89F0FF54/9AEBA7578B594B838FE7C04D58D4258F/TI-PyAppPrgG_v570_EN.pdf) for more programming information.