Python Syntax Reference:

For more on programming Rover with TI-Nspire CXII follow the links to the TI Rover Menu Map: TI-Nspire™ Python Programming > Python Menu Map > TI Rover Menu

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 Rover as the origin and the heading as 0 degrees measured from the x-axis.
import module_name as name_space	`import ti_plotlib as plt`	Required for the **text_at()** function. Imports the ti_plotlib module into the Python program. The module provides the methods for displaying text and data plots. The import ti_plotlib as plt statement is available from the TI Plotlib menu.
from module_name import *	`from time import *`	Imports all the functions in the **time** module for use in the program. It is necessary to import the time module to use the **sleep()** function. The from time import * statement is available from the More Modules>Time 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 the **while get_key() !=** “esc” statement. The from ti_system import * statement is available from the More Modules>TI System 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.forward(distance,”unit”,speed,”rate”)	`rv.forward(5,”units”,1.5,”units/s”)`	Rover drives 5 units forward at 1.5 units per second. The default unit is 10 cm. The default speed is 2 units pers second.
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 Rover 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 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 TI Rover Path menu (look toward the bottom of the menu)

Statement	Example	Behavior
text_at(row,”text”,”align”)	`text_at(3,“x position = "+str(x),"left")`	The **text_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 Note: The **str()** function converts a numeric value to a string. The **+** operator is used to join two strings. **str()** is available from the Built-ins> Type menu.
get_key()	`key_pressed=get_key()`	get_key() is a function that returns a string with the value associated with the last key pressed while a program is running. The value of the escape key is “esc”. In the example, pressing the escape key updates the variable **key_pressed** to “esc”.

while get_key() != “esc”:
block
	`while get_key() != “esc”:`
	`  x=rv.waypoint_x()`
	`  text_at(3,"x position ="+str(x),"left")`
		Defines a while loop that will continue until the escape 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 escape key is pressed. Not pressing a key or pressing any key but escape means that get_key() will return a value that is not equal to “esc”. The loop condition is true and looping continues. If the escape key is pressed, get_key() returns “esc”. The condition will evaluate as “esc” not equal to “esc”, which is false. A false result means that the loop statements are not repeated. 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 get_key() != “esc”: is available from the TI Hub > Commands menu.

Statement	Example	Behavior

<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. The examples show some of the relational operators available from the Built-ins Ops menu.
		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 Built-in Ops menu or from the menu brought up by pressing ctrl [=] on the TI-Nspire keyboard.

if <Boolean expression>:
block
	`if 0<x<2:`
	`  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:`
	`  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.