Meet TI-Rover
Geometry Challenges Day

TI-84 Plus CE
Python

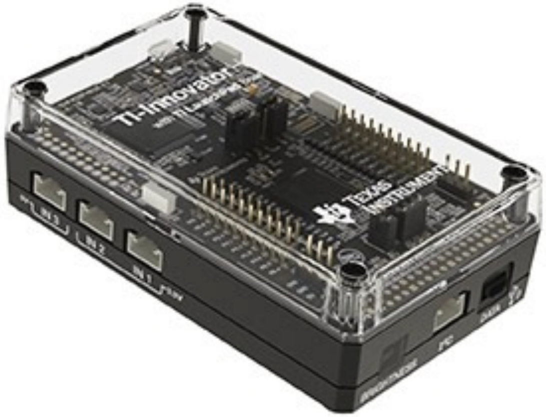
Texas Instruments

@ticalculators

Meet the TI-Innovator™ Rover

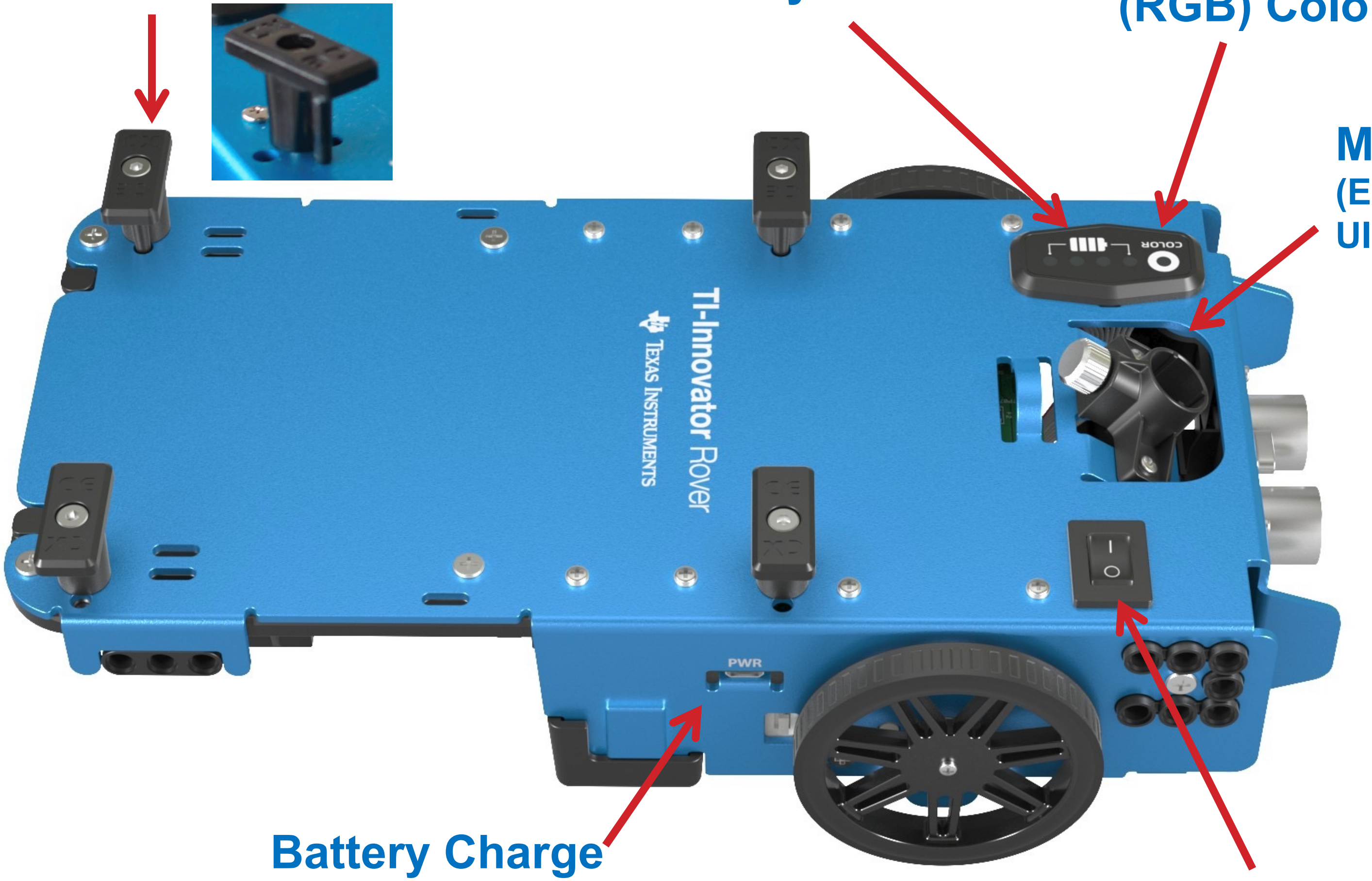
TI Graphing Calculator

TI-Innovator™ Rover



TI-Innovator™ Hub

Rover from the top



Calculator holder posts.
Lift and twist to CE or CX side.

Battery indicator

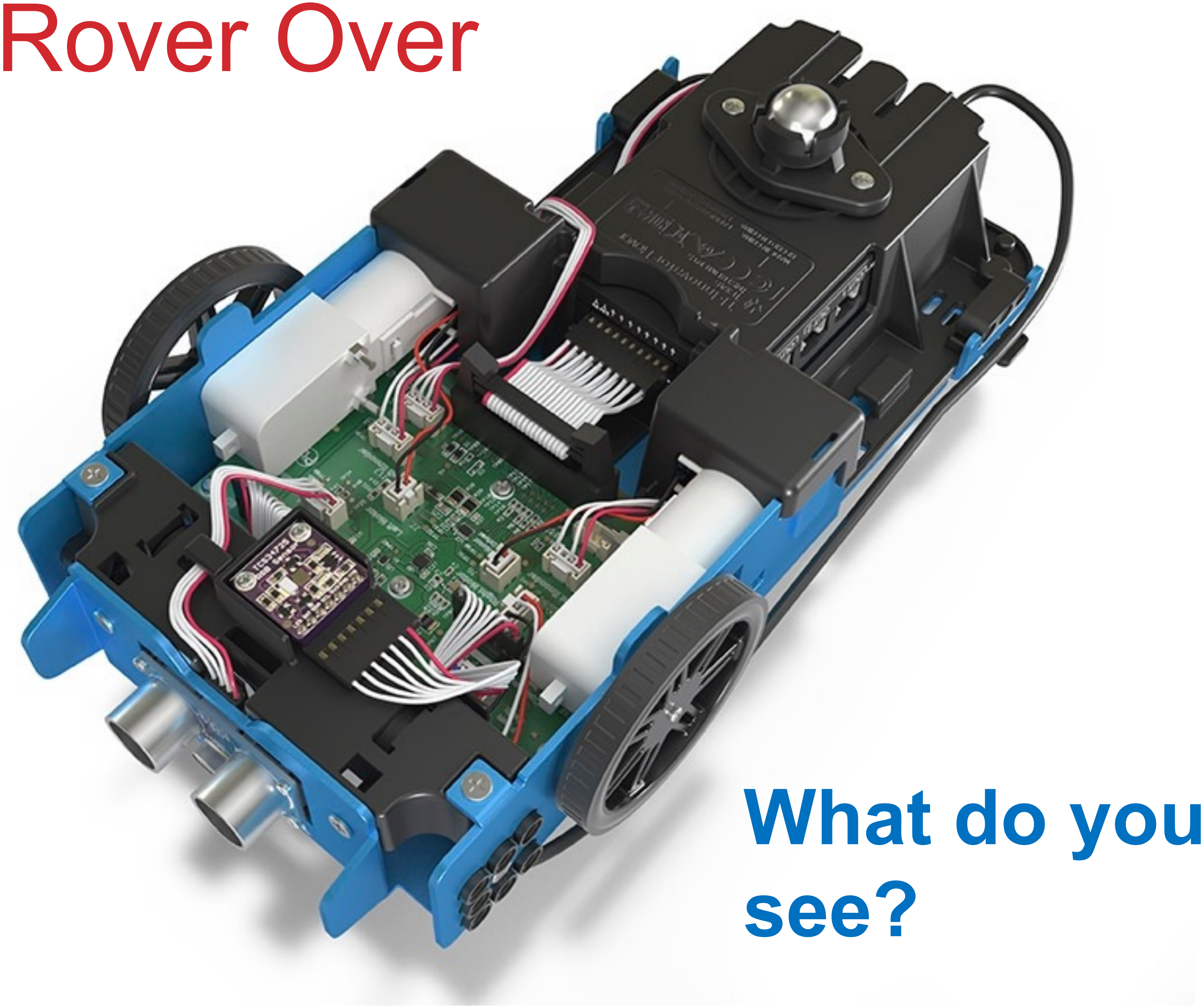
Red-Green-Blue (RGB) Color LED

Marker holder
(Expo Fine and Ultra Fine sizes)

Battery Charge with USB micro to wall adapter.

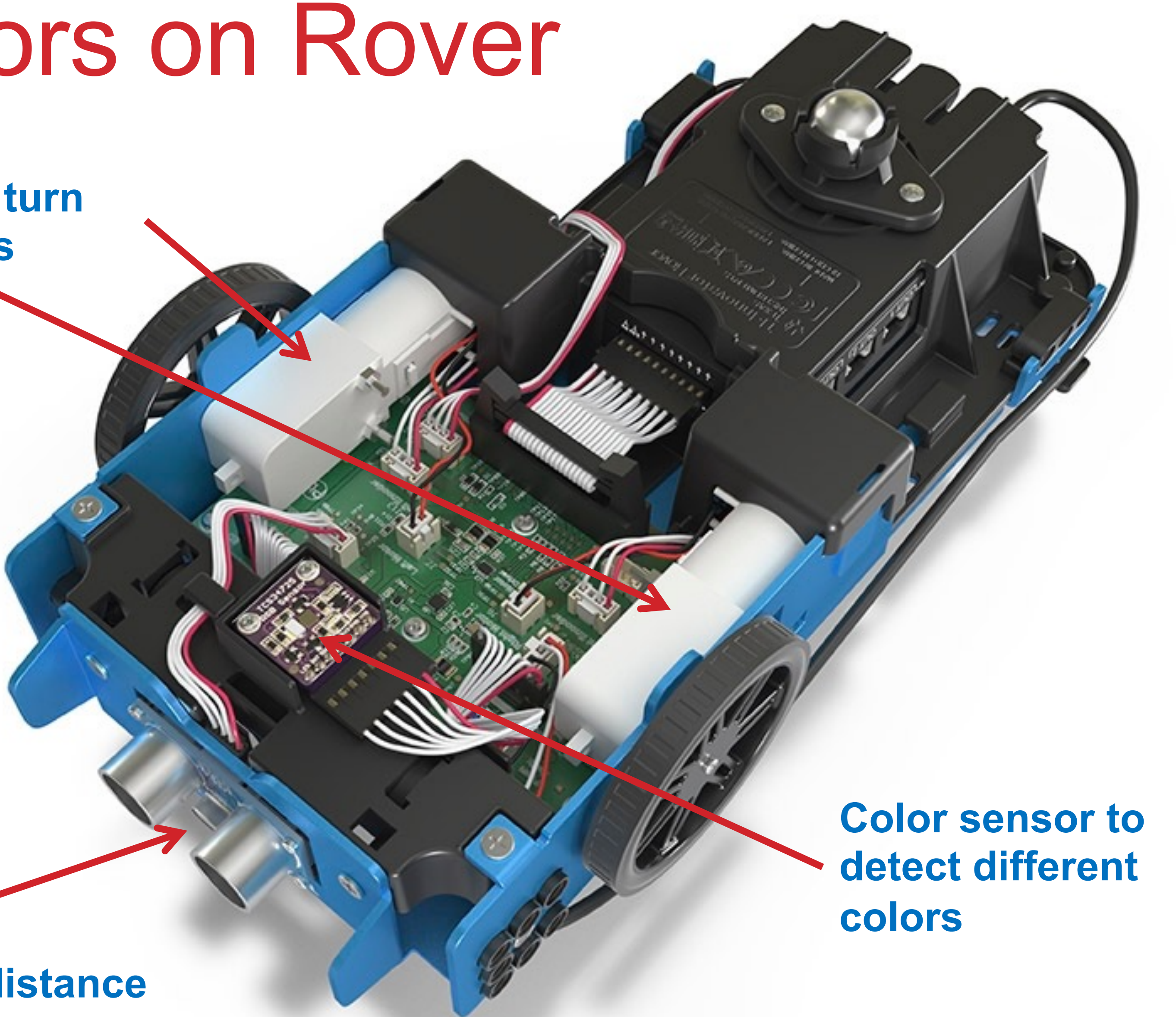
On/Off Switch

Turn Rover Over



What do you see?

Sensors on Rover

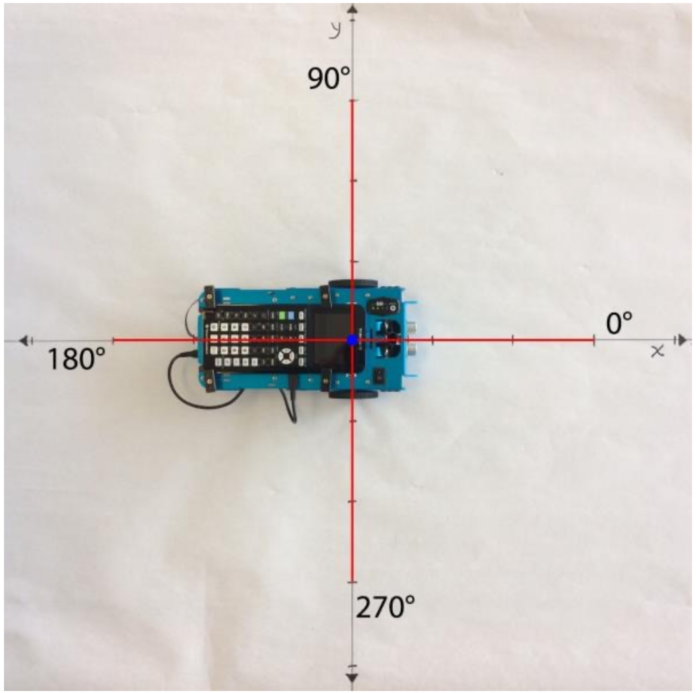


Motors to turn the wheels

Color sensor to detect different colors

Ranger to measure distance

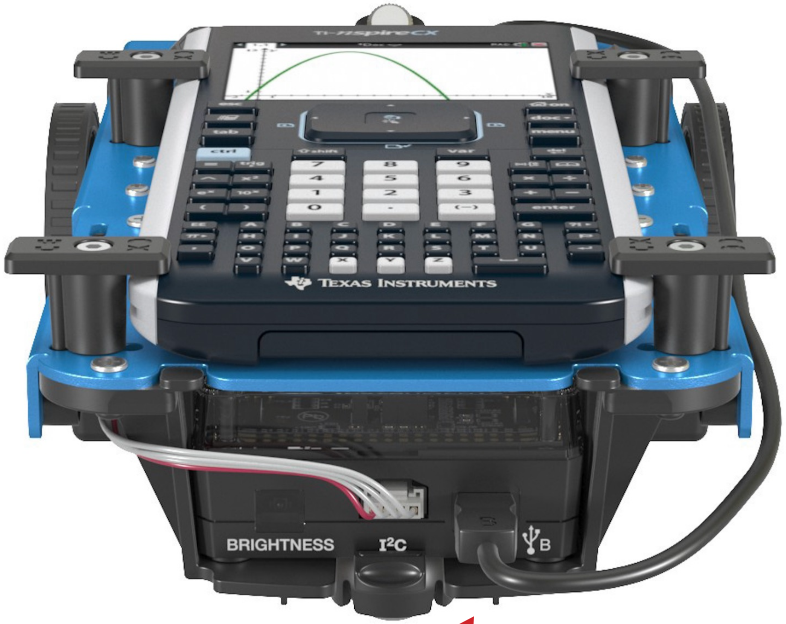
TI-Rover orientation and virtual grid



Rover programs set the initial position as the origin and the heading as 0 degrees measured from the x-axis.

Note: The Rover tracks its position on a virtual coordinate grid with a unit value of 10 cm. The coordinate grid position applies to the to_xy(x,y), to_polar(r,theta_degrees) and to_angle(angle, “unit”) functions on the Rover Drive menu. The virtual grid also applies to Path menu functions.

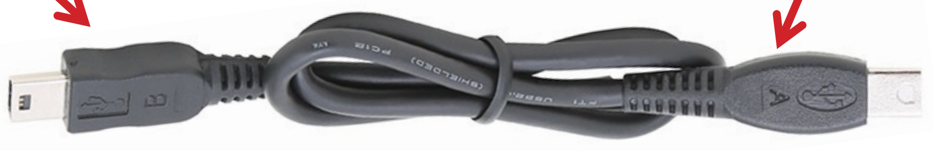
Connecting Rover to your calculator



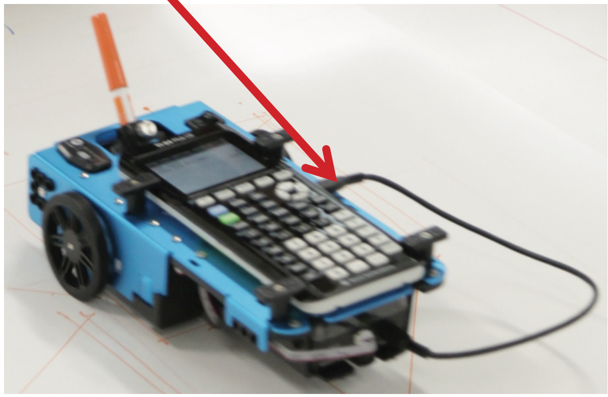
1 Make sure that your Rover is switched on.

3 Plug A side into port on calculator the Rover Hub.

2 Plug B side into USB B port of the Rover Hub.

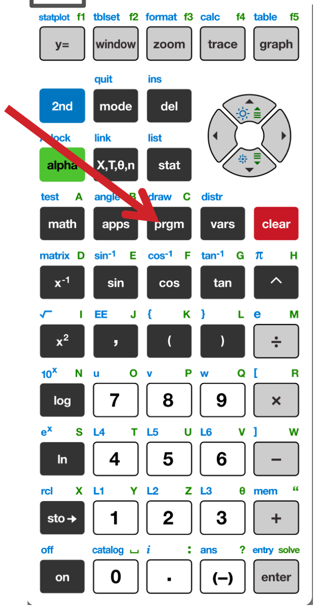


Unit-to-unit cable



Creating a new Python Program

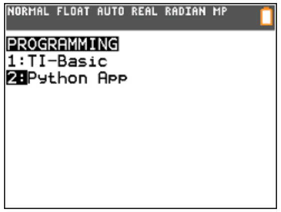
1



statplot f1 tblset f2 format f3 calc f4 table f5
y= window zoom trace graph
quit ins
2nd mode del
A-lock link list
alpha X,T,θ,n stat
test A angle B draw C distr
math apps prgm vars clear
matrix D sin⁻¹ E cos⁻¹ F tan⁻¹ G π H
x⁻¹ sin cos tan ^
√ I EE J { K } L e M
x² , ( ) ÷
10ˣ N u O v P w Q [R
log 7 8 9 ×
eˣ S L4 T L5 U L6 V] W
ln 4 5 6 −
rcl X L1 Y L2 Z L3 θ mem "
sto→ 1 2 3 +
off catalog ␣ i : ans ? entry solve
on 0 . (−) enter

Press the **[prgm]** key to create, edit and execute TI-Python programs.

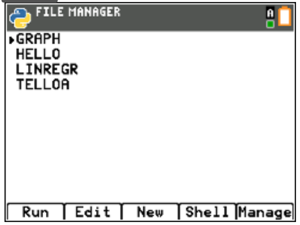
2



```
NORMAL FLOAT AUTO REAL RADIAN MP
PROGRAMMING
1:TI-Basic
2:Python App
```

Press **[enter]** or Press **[2]** to select 2: Python App

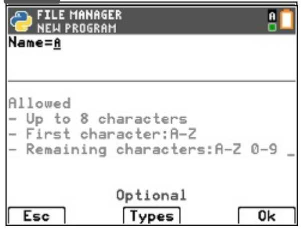
3



```
FILE MANAGER
▸GRAPH
 HELLO
 LINREGR
 TELLOA
Run | Edit | New | Shell | Manage
```

You have the option to run, edit, create or manage programs.

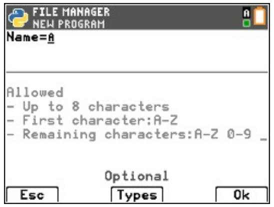
4



```
FILE MANAGER
NEW PROGRAM
Name=A

Allowed
- Up to 8 characters
- First character:A-Z
- Remaining characters:A-Z 0-9 _

Optional
Esc        Types        Ok
```

Press **[New]** softkey (trace button)

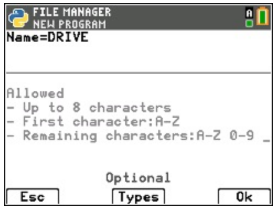
5



```
FILE MANAGER
NEW PROGRAM
Name=A

Allowed
- Up to 8 characters
- First character:A-Z
- Remaining characters:A-Z 0-9 _

Optional
Esc        Types        Ok
```

You are prompted to enter a program name. The blinking A cursor shows that you are in alpha entry mode. The green alpha labels on the keys are active.

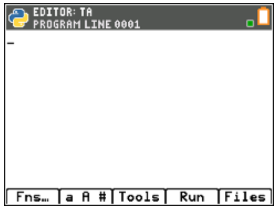
6



```
FILE MANAGER
NEW PROGRAM
Name=DRIVE

Allowed
- Up to 8 characters
- First character:A-Z
- Remaining characters:A-Z 0-9 _

Optional
Esc        Types        Ok
```

Type your program name and press **[Ok].**

7



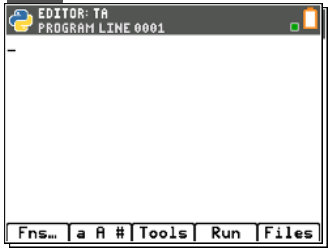
```
EDITOR: TA
PROGRAM LINE 0001
_

Fns… | a A # | Tools | Run | Files
```

You are now in position to begin entering statements to your program.

Entering a TI-Rover Program –

importing the TI-Rover module and connecting to a Rover

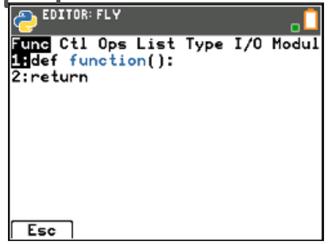
1



```
EDITOR: TA
PROGRAM LINE 0001
_
```

The Python program editor uses an insert cursor and a backspace delete.
Press **[Fns…]** softkey to see functions to use in your program.

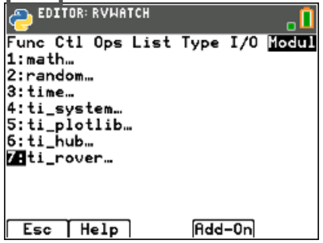
2



```
EDITOR: FLY
1:def function():
2:return
```

Press **right arrow** repeatedly or **left arrow** to move to the Modul menu.

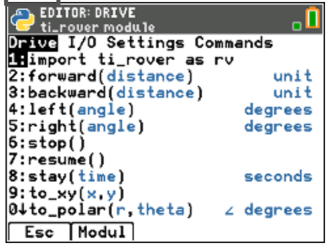
3



```
EDITOR: RVWATCH
1:math…
2:random…
3:time…
4:ti_system…
5:ti_plotlib…
6:ti_hub…
7:ti_rover…
```

You will see a menu of installed modules available to use functions from. Select **7:ti_rover**.

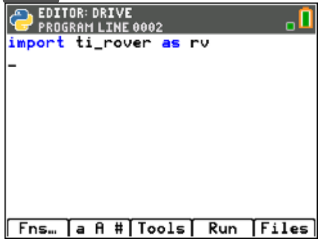
4



```
EDITOR: DRIVE
ti_rover module
1:import ti_rover as rv
2:forward(distance)          unit
3:backward(distance)         unit
4:left(angle)             degrees
5:right(angle)            degrees
6:stop()
7:resume()
8:stay(time)              seconds
9:to_xy(x,y)
0↓to_polar(r,theta)     ∠ degrees
```

Select **1:import ti_rover as rv**.

5

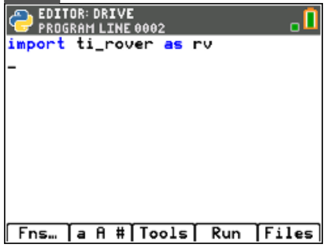


```
EDITOR: DRIVE
PROGRAM LINE 0002
import ti_rover as rv
_
```

The ti_rover module import statement is pasted to your program. The ti_rover import statement is required at the beginning of every Rover program. This import statement brings in Rover functions to use in your program, sets Rover's initial position and sets up communication between the Rover and the Hub.

Entering a TI-Rover Program

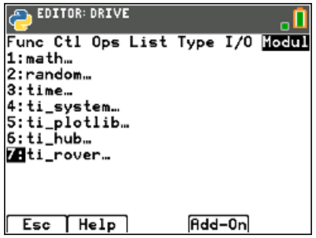
1



```
EDITOR: DRIVE
PROGRAM LINE 0002
import ti_rover as rv
_
```

You are now ready to enter functions to control your Rover. Navigate to the Rover menus by pressing **[Fns…]** then **arrow** to the Modul menu.

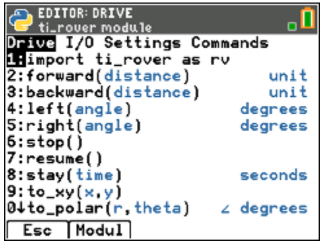
2



```
EDITOR: DRIVE
Func Ctl Ops List Type I/O Modul
1:math…
2:random…
3:time…
4:ti_system…
5:ti_plotlib…
6:ti_hub…
7:ti_rover…
```

Then **select ti_rover...** to see options.

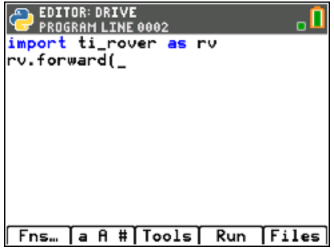
3



```
EDITOR: DRIVE
ti_rover module
Drive I/O Settings Commands
1:import ti_rover as rv
2:forward(distance)            unit
3:backward(distance)           unit
4:left(angle)               degrees
5:right(angle)              degrees
6:stop()
7:resume()
8:stay(time)                seconds
9:to_xy(x,y)
0↓to_polar(r,theta)       ∠ degrees
```

You begin on the Drive menu. **Select** the **2:forward()** function.

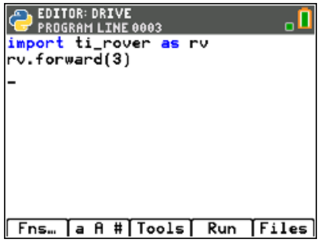
4



```
EDITOR: DRIVE
PROGRAM LINE 0002
import ti_rover as rv
rv.forward(_
```

Enter a value for the number of Rover units to drive forward. **Arrow to the end of the statement** and press **[enter]** to move to the next statement.

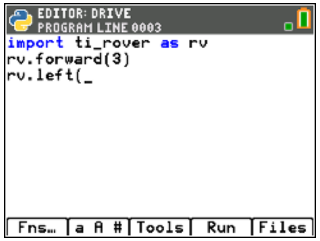
5



```
EDITOR: DRIVE
PROGRAM LINE 0003
import ti_rover as rv
rv.forward(3)
_
```

Navigate to the Drive menu again by press **[fns…]**, **left arrow**, **7:ti_rover…,4:left()** to select the **left turn function**.

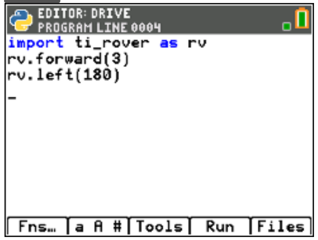
6



```
EDITOR: DRIVE
PROGRAM LINE 0003
import ti_rover as rv
rv.forward(3)
rv.left(_
```

Enter a value for the angle to turn in degrees. **Arrow to the end of the statement** and press **[enter]** to move to the next statement.

7

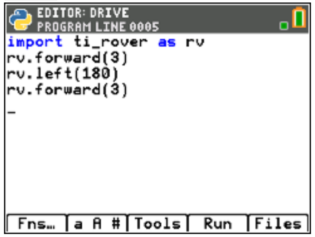


```
EDITOR: DRIVE
PROGRAM LINE 0004
import ti_rover as rv
rv.forward(3)
rv.left(180)
_
```

Navigate to the Drive menu again, then select **2:forward()**. After the function is pasted **enter the Rover units** to drive. **Arrow to the end of the statement** and press **[enter]** to move to the next statement.

.

8



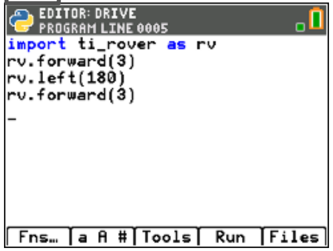
```
EDITOR: DRIVE
PROGRAM LINE 0005
import ti_rover as rv
rv.forward(3)
rv.left(180)
rv.forward(3)
_
```

You are now ready to run your TI-Rover program.

.

Running a TI-Rover Program

1



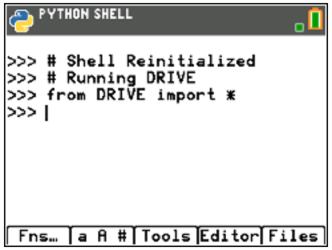
```
import ti_rover as rv
rv.forward(3)
rv.left(180)
rv.forward(3)
_
```

You are now ready to run your program.

Before pressing **[Run]** go through the pre-drive checklist.

1. Make sure that TI-Rover is turned ON.
2. Make sure that the calculator unit-to-unit cable is connected to the Hub inside the Rover. Plug the B end of the cable into the Data USB B port of the Hub. Plug the A end of the cable into the calculator.
3. Press **[Run]**.

2

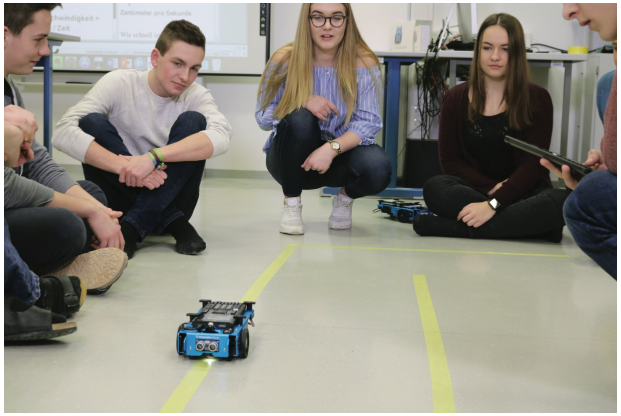


```
>>> # Shell Reinitialized
>>> # Running DRIVE
>>> from DRIVE import *
>>> |
```

The program will run in the Python shell. You will receive messages on the status of the program.

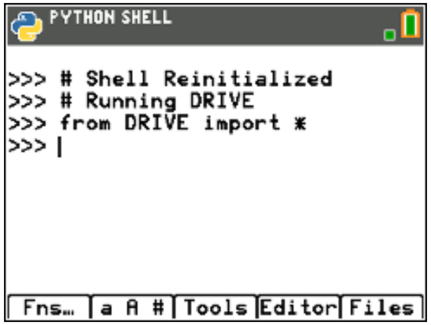
You can run the program again by pressing **[Tools]** and selecting **1:Rerun Last Program** from the menu.

You can return to the program editor by pressing **[Editor]**.



Editing a Rover Program

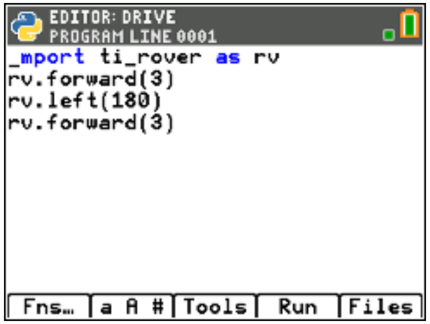
1



```
>>> # Shell Reinitialized
>>> # Running DRIVE
>>> from DRIVE import *
>>> |
```

Press **[Editor]** to go back to your Python editor page.

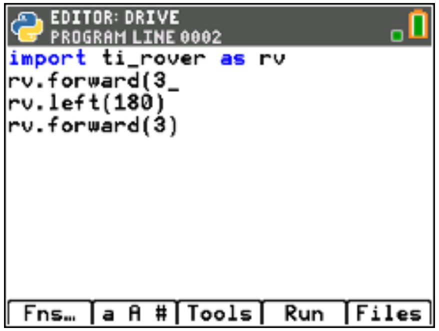
2



```
_mport ti_rover as rv
rv.forward(3)
rv.left(180)
rv.forward(3)
```

Use the arrow keys to position the cursor to change the value of the forward distance.

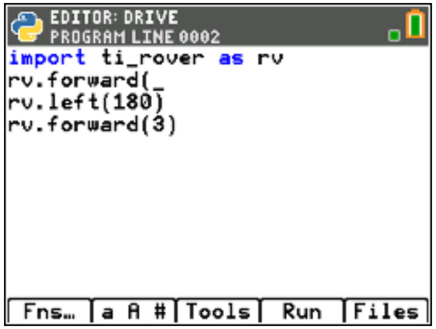
3



```
import ti_rover as rv
rv.forward(3_
rv.left(180)
rv.forward(3)
```

Press **[del]** to backspace over the 3.

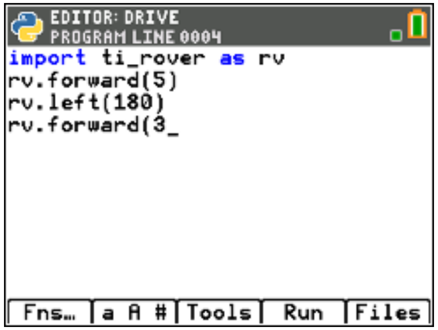
4



```
import ti_rover as rv
rv.forward(_
rv.left(180)
rv.forward(3)
```

Type in a new value for distance, **right arrow** to the end of the line, then **down arrow** to position the cursor to change the value of the second forward() function.

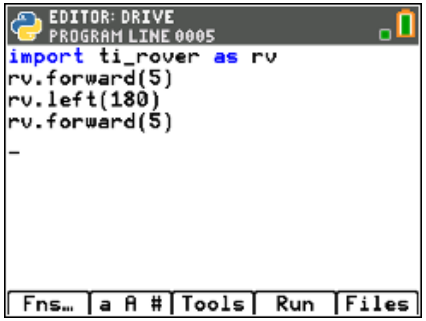
5



```
import ti_rover as rv
rv.forward(5)
rv.left(180)
rv.forward(3_
```

Press **[del]** to backspace over the current distance value. **Type in a new value** for distance, **right arrow** to the end of the line, then **[enter]** to move to a new statement.

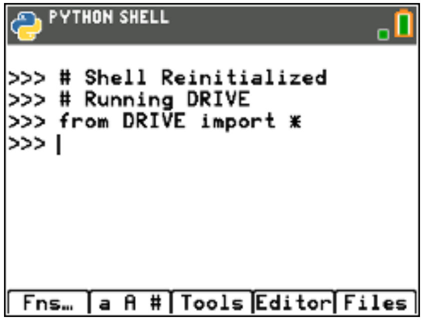
6



```
import ti_rover as rv
rv.forward(5)
rv.left(180)
rv.forward(5)
_
```

Press **[Run]** to run the program in the Python shell.

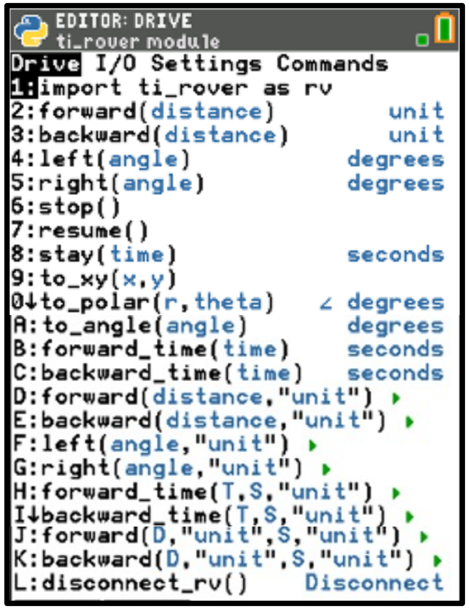
7



```
>>> # Shell Reinitialized
>>> # Running DRIVE
>>> from DRIVE import *
>>> |
```

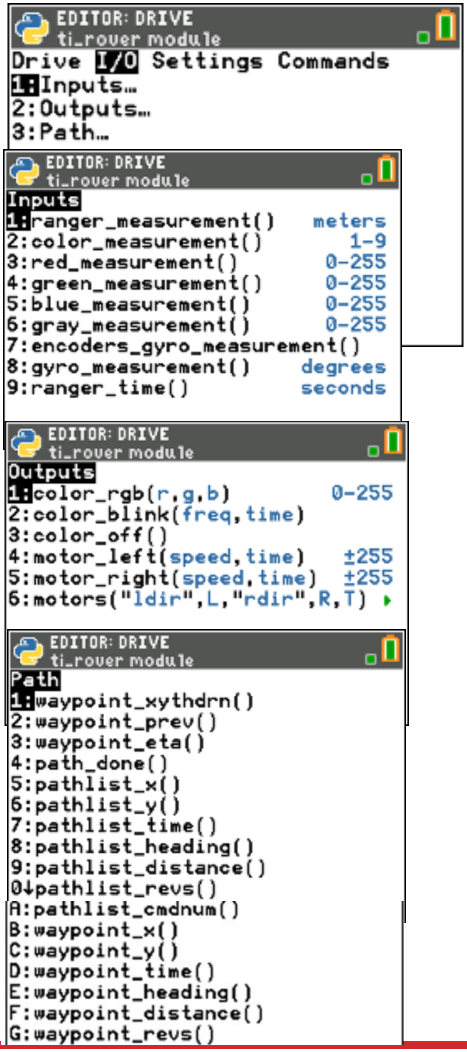
TI-Rover Module Menus

Drive



```
EDITOR: DRIVE
ti_rover module
Drive I/O Settings Commands
1:import ti_rover as rv
2:forward(distance)           unit
3:backward(distance)          unit
4:left(angle)              degrees
5:right(angle)             degrees
6:stop()
7:resume()
8:stay(time)               seconds
9:to_xy(x,y)
0↓to_polar(r,theta)      ∠ degrees
A:to_angle(angle)          degrees
B:forward_time(time)       seconds
C:backward_time(time)      seconds
D:forward(distance,"unit") ▸
E:backward(distance,"unit") ▸
F:left(angle,"unit") ▸
G:right(angle,"unit") ▸
H:forward_time(T,S,"unit") ▸
I↓backward_time(T,S,"unit") ▸
J:forward(D,"unit",S,"unit") ▸
K:backward(D,"unit",S,"unit") ▸
L:disconnect_rv()       Disconnect
```

Input/Output (I/O)



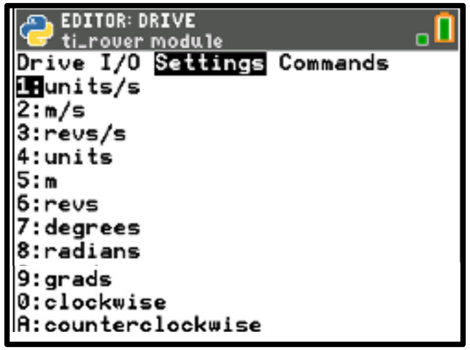
```
EDITOR: DRIVE
ti_rover module
Drive I/O Settings Commands
1:Inputs…
2:Outputs…
3:Path…
```

```
EDITOR: DRIVE
ti_rover module
Inputs
1:ranger_measurement()      meters
2:color_measurement()          1-9
3:red_measurement()          0-255
4:green_measurement()        0-255
5:blue_measurement()         0-255
6:gray_measurement()         0-255
7:encoders_gyro_measurement()
8:gyro_measurement()       degrees
9:ranger_time()            seconds
```

```
EDITOR: DRIVE
ti_rover module
Outputs
1:color_rgb(r,g,b)           0-255
2:color_blink(freq,time)
3:color_off()
4:motor_left(speed,time)      ±255
5:motor_right(speed,time)     ±255
6:motors("ldir",L,"rdir",R,T) ▸
```

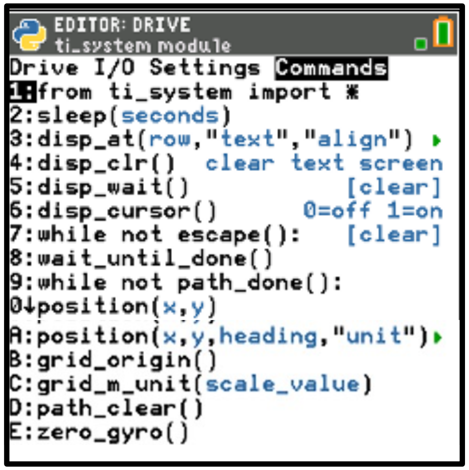
```
EDITOR: DRIVE
ti_rover module
Path
1:waypoint_xythdrn()
2:waypoint_prev()
3:waypoint_eta()
4:path_done()
5:pathlist_x()
6:pathlist_y()
7:pathlist_time()
8:pathlist_heading()
9:pathlist_distance()
0↓pathlist_revs()
A:pathlist_cmdnum()
B:waypoint_x()
C:waypoint_y()
D:waypoint_time()
E:waypoint_heading()
F:waypoint_distance()
G:waypoint_revs()
```

Settings



```
EDITOR: DRIVE
ti_rover module
Drive I/O Settings Commands
1:units/s
2:m/s
3:revs/s
4:units
5:m
6:revs
7:degrees
8:radians
9:grads
0:clockwise
A:counterclockwise
```

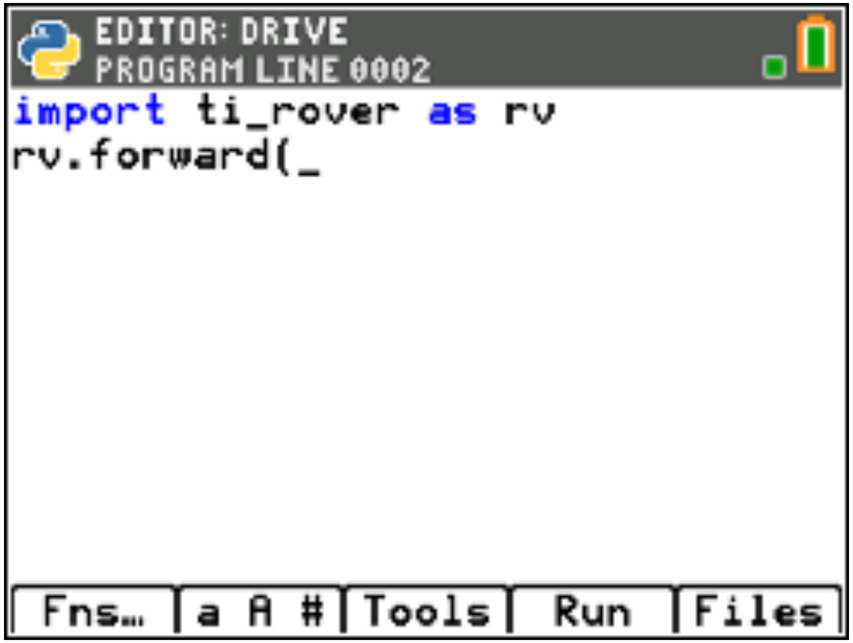
Commands



```
EDITOR: DRIVE
ti_system module
Drive I/O Settings Commands
1:from ti_system import *
2:sleep(seconds)
3:disp_at(row,"text","align") ▸
4:disp_clr()  clear text screen
5:disp_wait()            [clear]
6:disp_cursor()      0=off 1=on
7:while not escape():    [clear]
8:wait_until_done()
9:while not path_done():
0↓position(x,y)
A:position(x,y,heading,"unit")▸
B:grid_origin()
C:grid_m_unit(scale_value)
D:path_clear()
E:zero_gyro()
```

MAKE IT MOVE!

New Program:



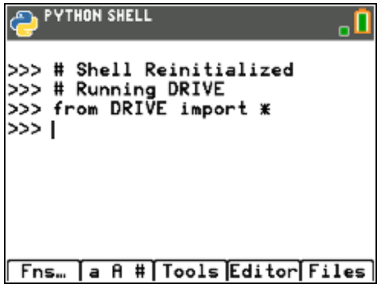
```
EDITOR: DRIVE
PROGRAM LINE 0002
import ti_rover as rv
rv.forward(_
```

Press **[Fns…], left arrow,** then **7:ti_rover…** for the Rover menus.

Press **[Run]** to run the program in the Python shell.

Task: Discover how far Rover drives per unit.

Use differing values (1-20) to determine what 1 Rover unit is.

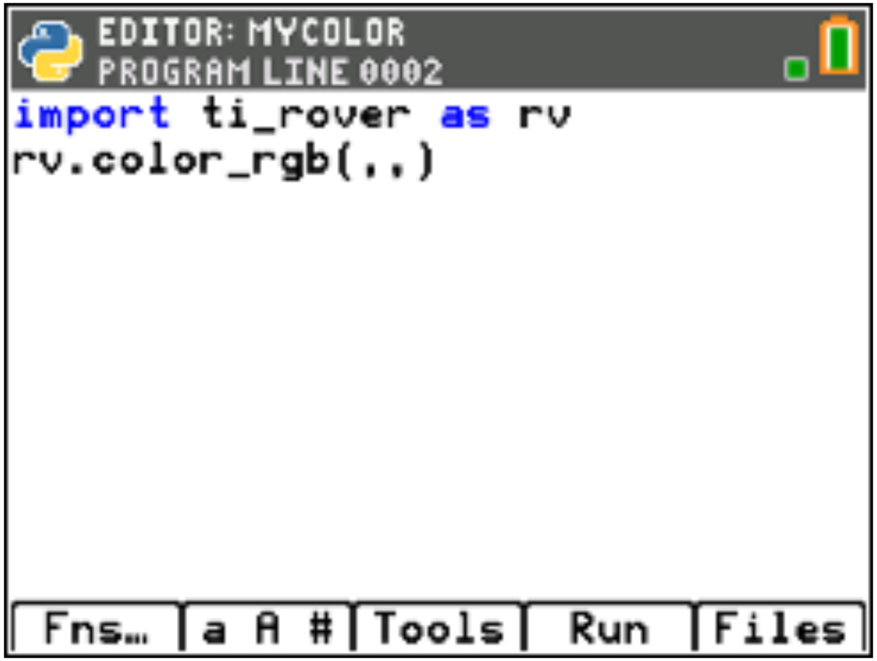


```
PYTHON SHELL
>>> # Shell Reinitialized
>>> # Running DRIVE
>>> from DRIVE import *
>>> |
```

From the Python shell, press **[Editor]** to move from the shell to the Python editor.

Set the color

New Program:



```
import ti_rover as rv
rv.color_rgb(,,)
```

Press **[Fns…], left arrow,** then **7:ti_rover…** for the Rover menus.

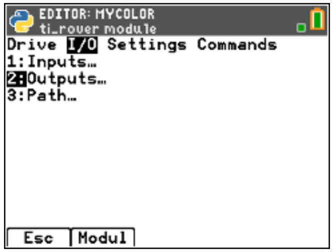
Press **[Run]** to run the program in the Python shell.

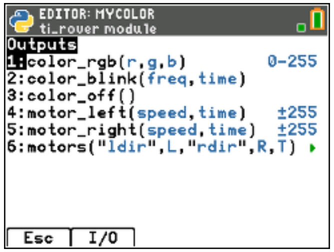
Task: Set the color output of the RGB LED.

Each color takes a value (0-255).

Challenge Task: Try to make Yellow

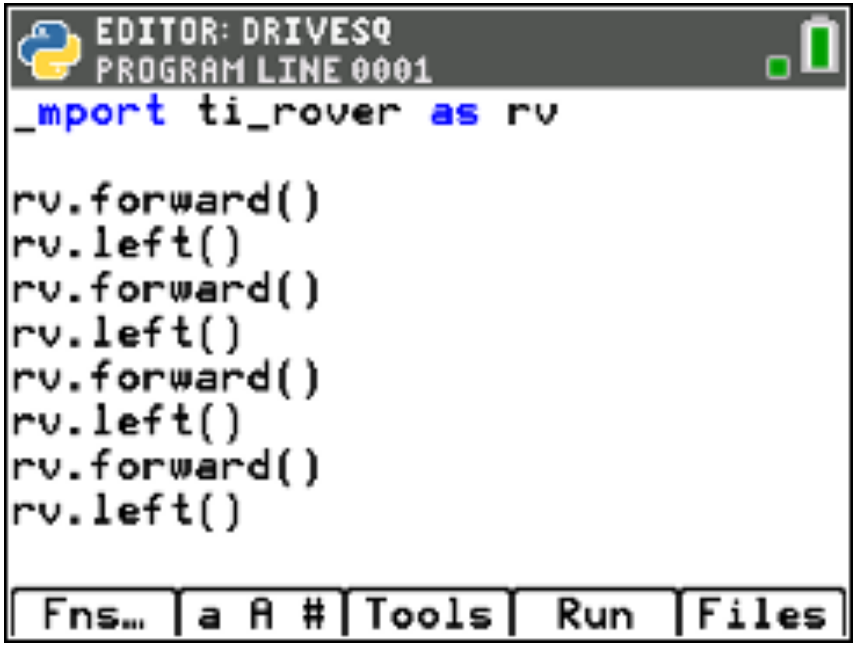
Find the color_rgb( ) function on the Rover Outputs menu. Enter values for the red, green and blue components of the color to display.





Explore angles

New Program:



```
_mport ti_rover as rv

rv.forward()
rv.left()
rv.forward()
rv.left()
rv.forward()
rv.left()
rv.forward()
rv.left()
```

The program above is a framework for driving a square.
Enter values for distance and turn angle.

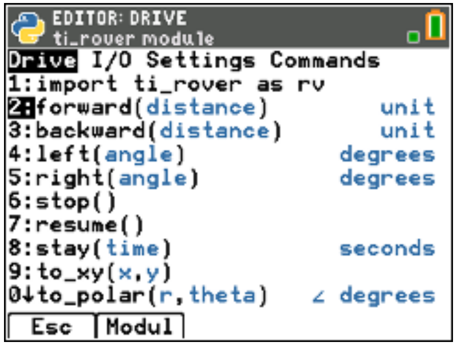
Press **[Fns…], left arrow,** then **7:ti_rover…** for the Rover menus.

Press **[Run]** to run the program in the Python shell.

Task: Drive a square.

Challenge Task: Try to drive an equilateral triangle.

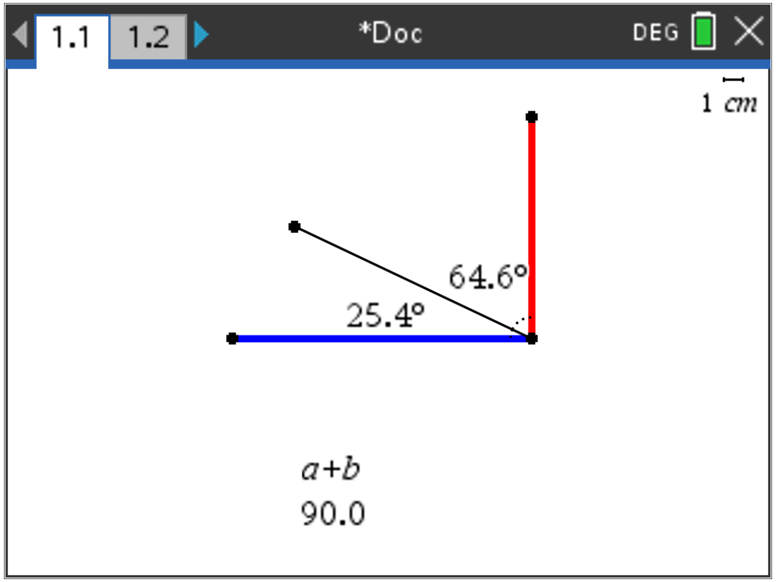
See the inputs for the most common drive functions below.



```
1:import ti_rover as rv
2:forward(distance)          unit
3:backward(distance)         unit
4:left(angle)             degrees
5:right(angle)            degrees
6:stop()
7:resume()
8:stay(time)              seconds
9:to_xy(x,y)
0↓to_polar(r,theta)     ∠ degrees
```

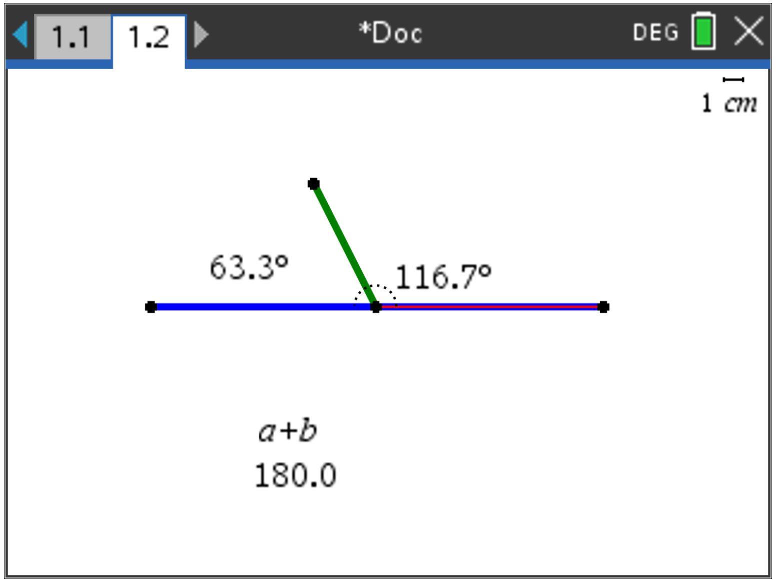
Quick Math Reminders

- Complementary Angles:
  - Sum to 90 degrees



a+b
90.0

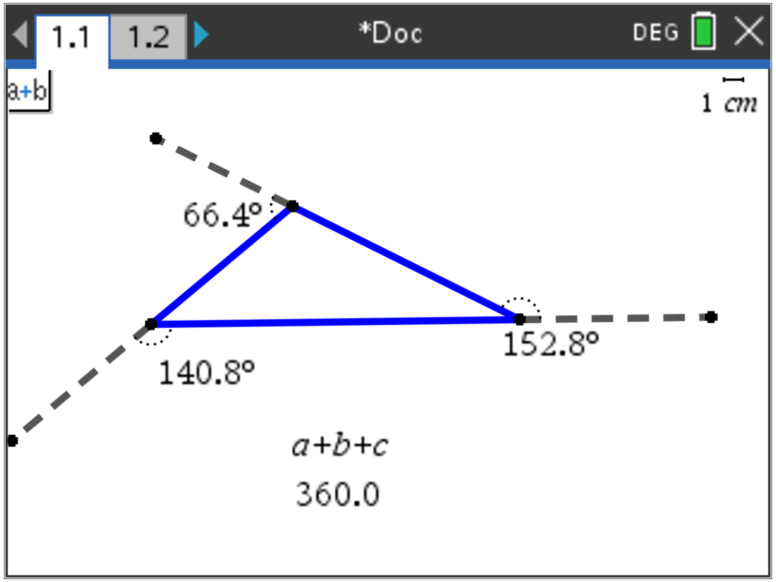
- Supplementary Angles:
  - Sum to 180 degrees

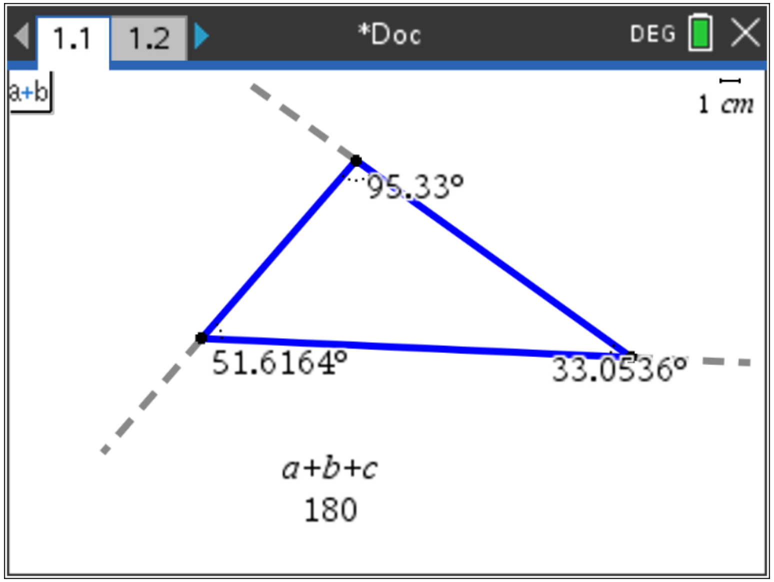


a+b
180.0

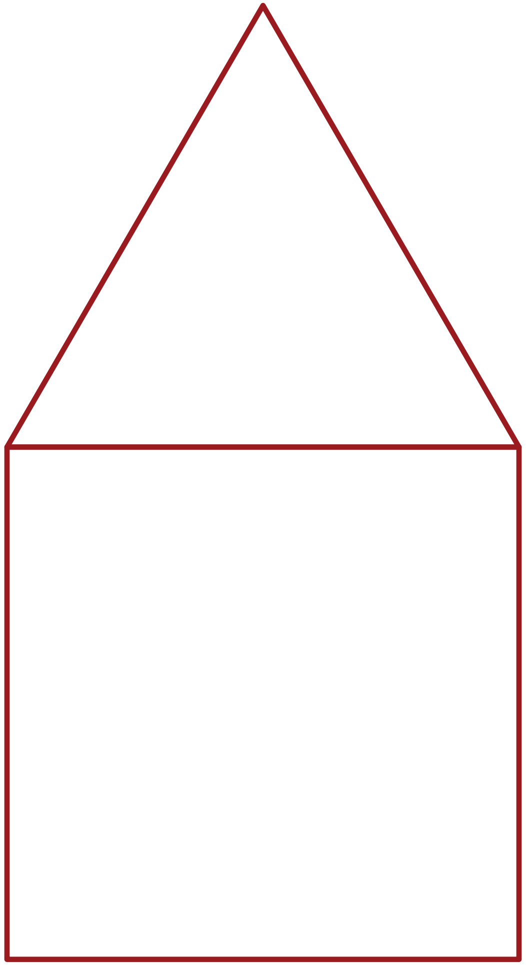
Quick Math Reminders

- » Exterior angles:
- » Interior Angles:



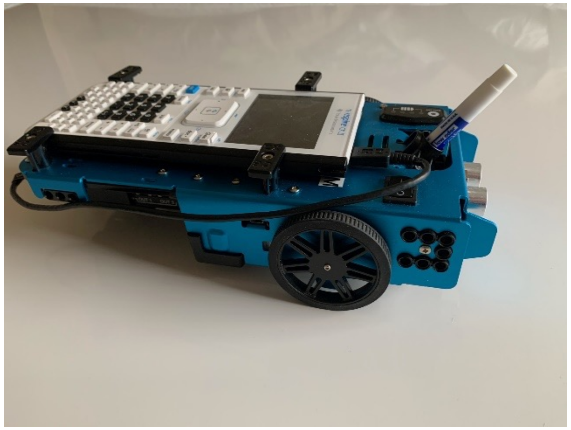


Logic Challenge

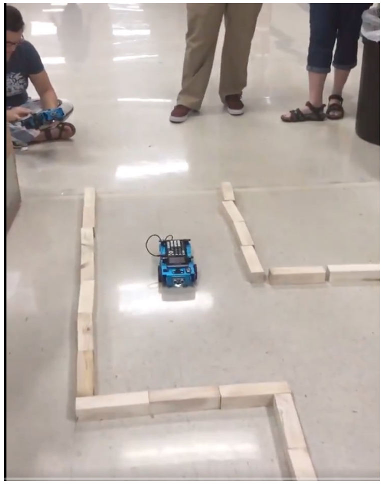


Task: Drive the figure shown without crossing any lines or going back over a line and without picking up the pen.

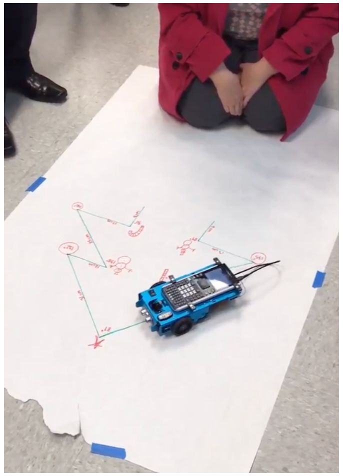
When you are ready put the pen in and trace your path



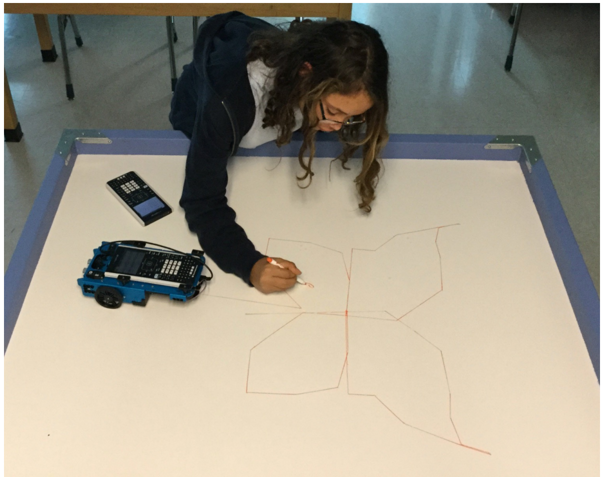
Where can you go next with TI-Rover?



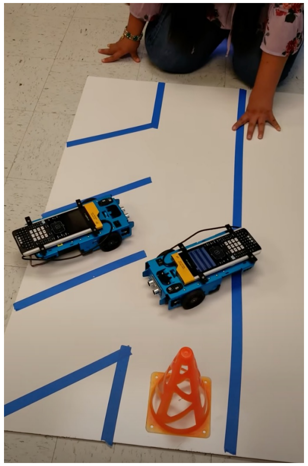
Drive an obstacle course



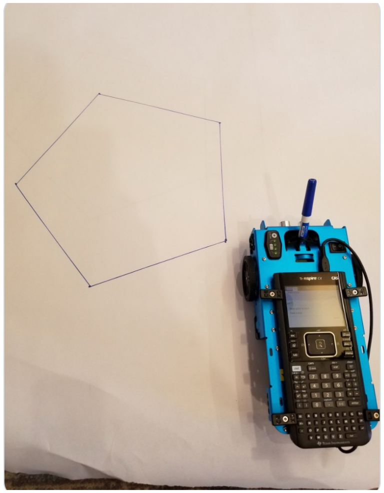
Drive a design



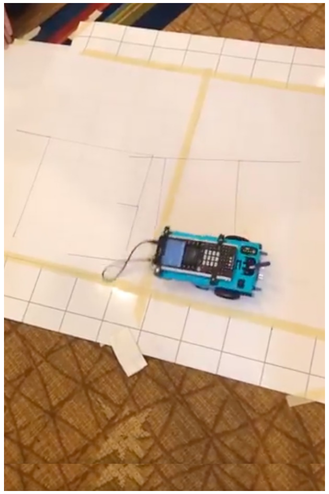
Draw artwork



Park your Rover



Use a For loop to draw polygons

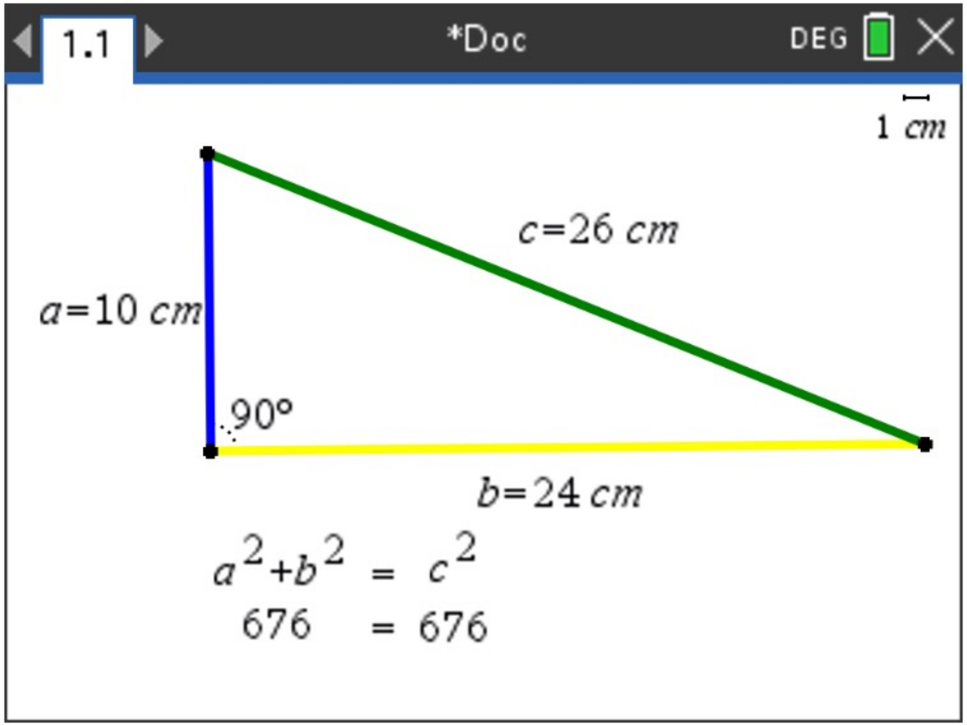


Write your name

Navigate a map

Quick Math Reminders

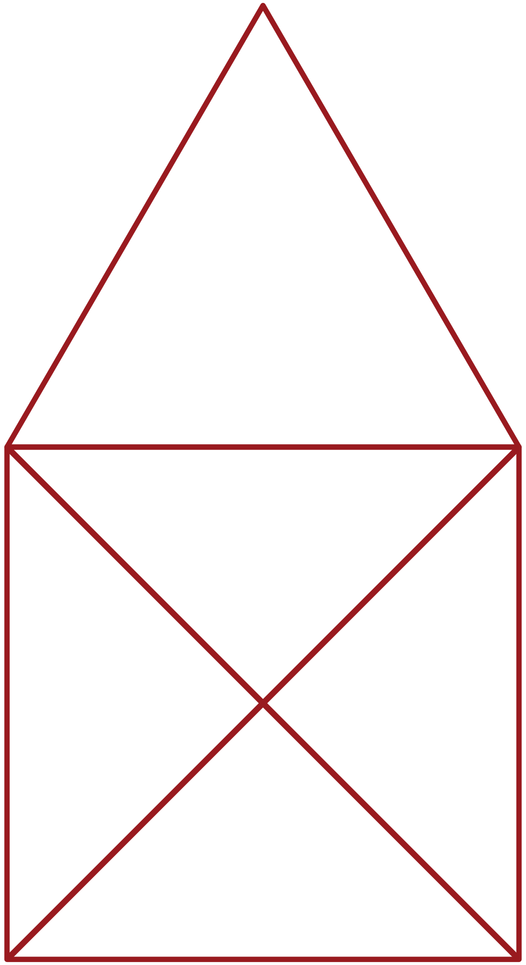
- » Pythagorean Theorem



$a^2+b^2 = c^2$

$676 = 676$

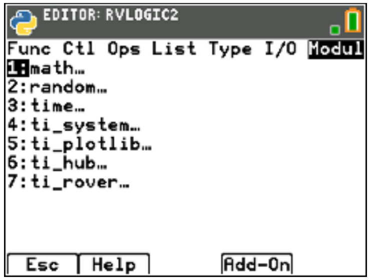
Logic Challenge 2

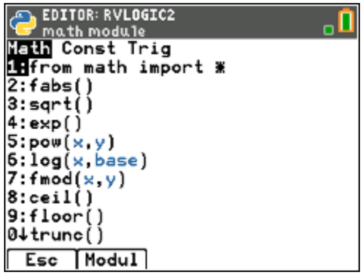


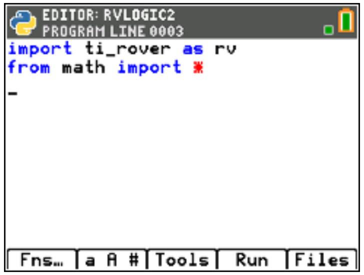
Task: Drive the figure shown without crossing any lines or going back over a line and without picking up the pen.

When you are ready put the pen in and trace your path

Import the Python Math module in addition to the Rover module for this challenge.

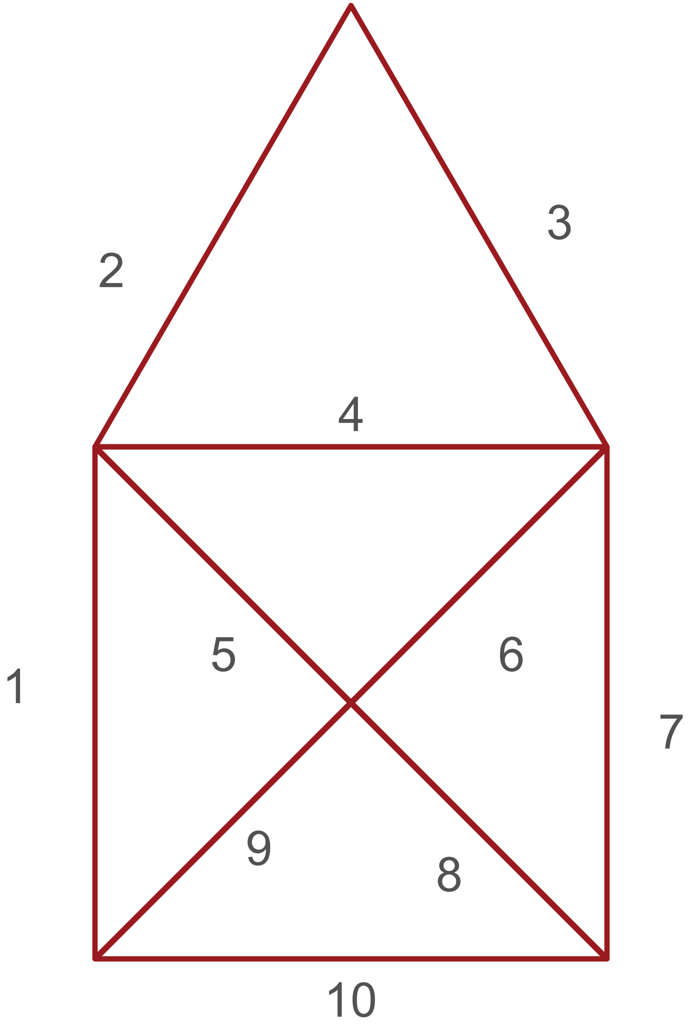


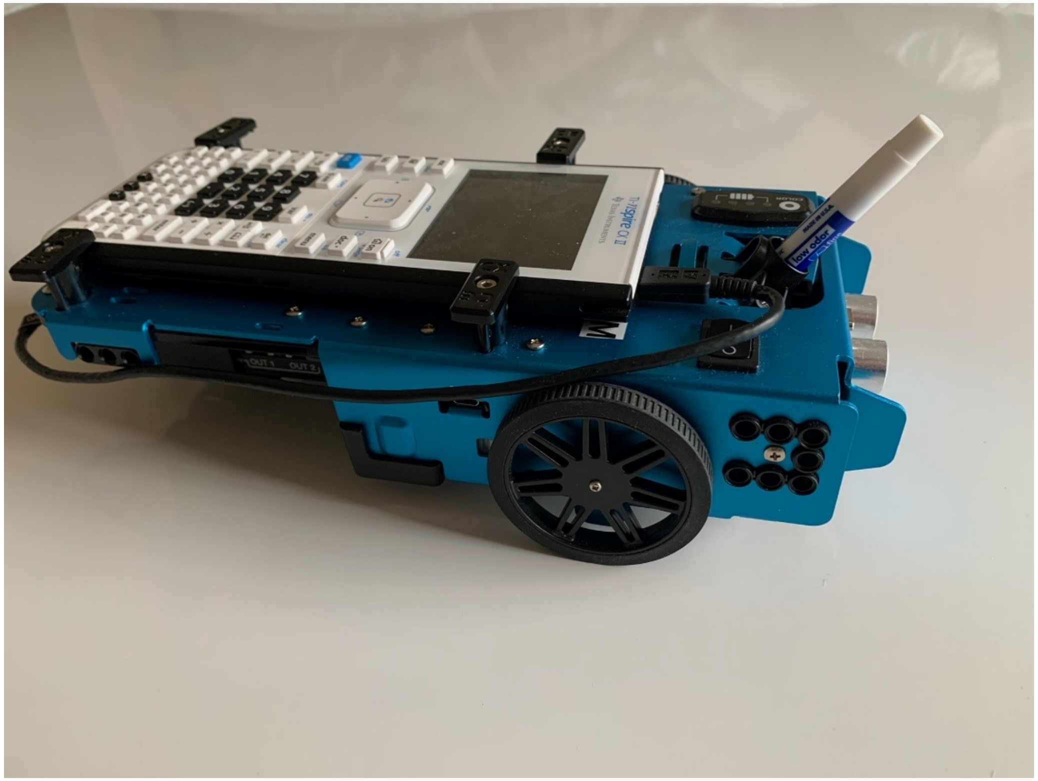




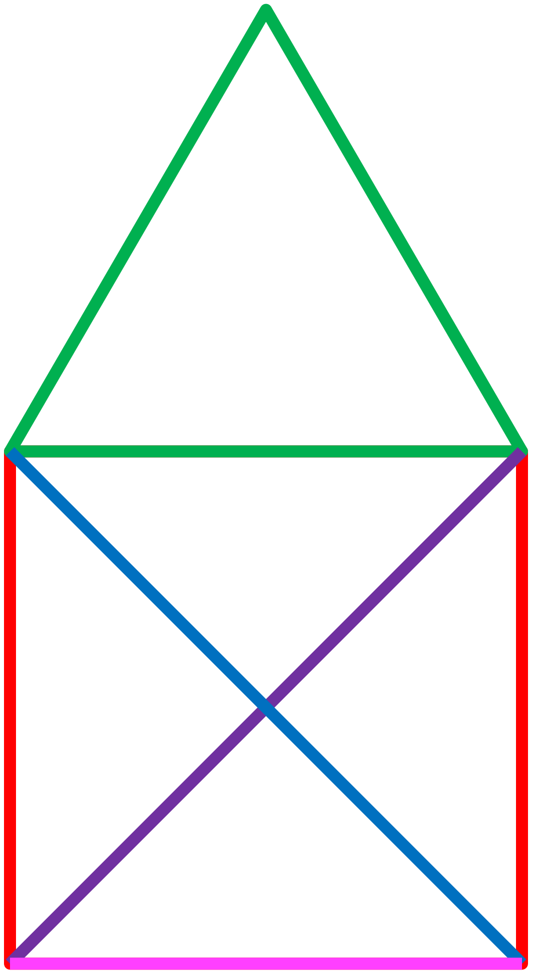
```
import ti_rover as rv
from math import *
```

Logic Challenge 2





Logic Challenge 3

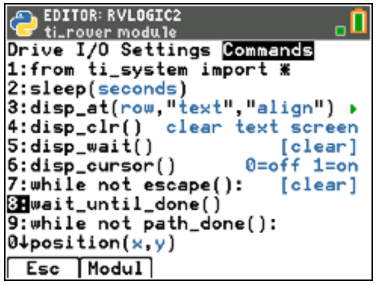


Task: Drive the figure shown without crossing any lines or going back over a line and without picking up the pen.

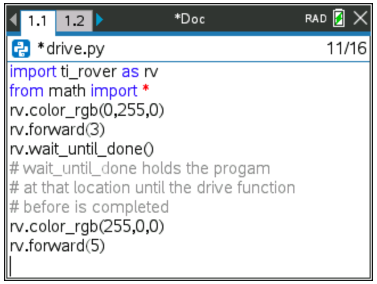
Now match the colors using the RGB LED. Don't worry about using the pen.

Import the Python Math module in addition to the Rover module for this challenge.

Use wait_until_done() from the Rover Commands menu to synchronize Rover drive functions with the RGB LED.



EDITOR: RVLOGIC2
ti_rover module
Drive I/O Settings Commands
1:from ti_system import *
2:sleep(seconds)
3:disp_at(row,"text","align")
4:disp_clr() clear text screen
5:disp_wait() [clear]
6:disp_cursor() 0=off 1=on
7:while not escape(): [clear]
8:wait_until_done()
9:while not path_done():
0↓position(x,y)
Esc
Modul



```
import ti_rover as rv
from math import *
rv.color_rgb(0,255,0)
rv.forward(3)
rv.wait_until_done()
# wait_until_done holds the progam
# at that location until the drive function
# before is completed
rv.color_rgb(255,0,0)
rv.forward(5)
```

Thank you!

See www.TIstemProjects.com for more TI STEM and coding activities and projects.