Meet
Wonder Workshop™ Dash with Geometry Challenges

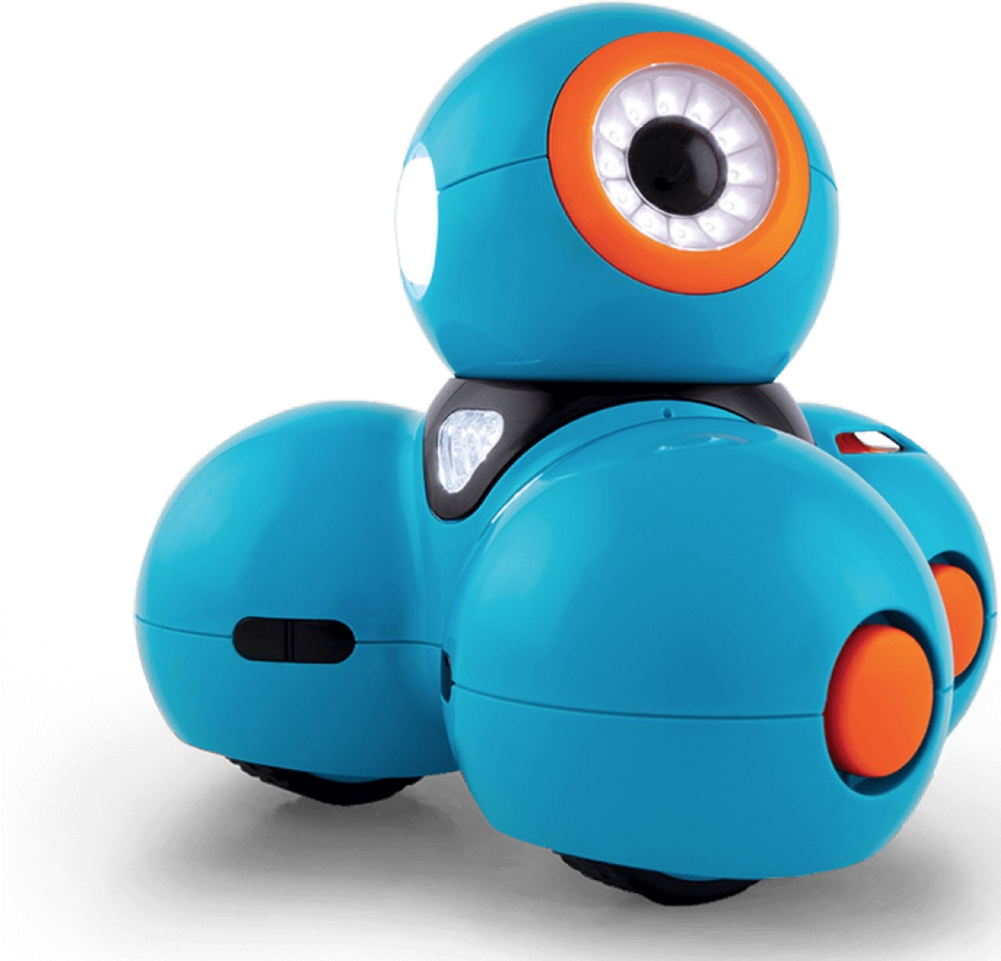
TI-84 Plus CE

Python

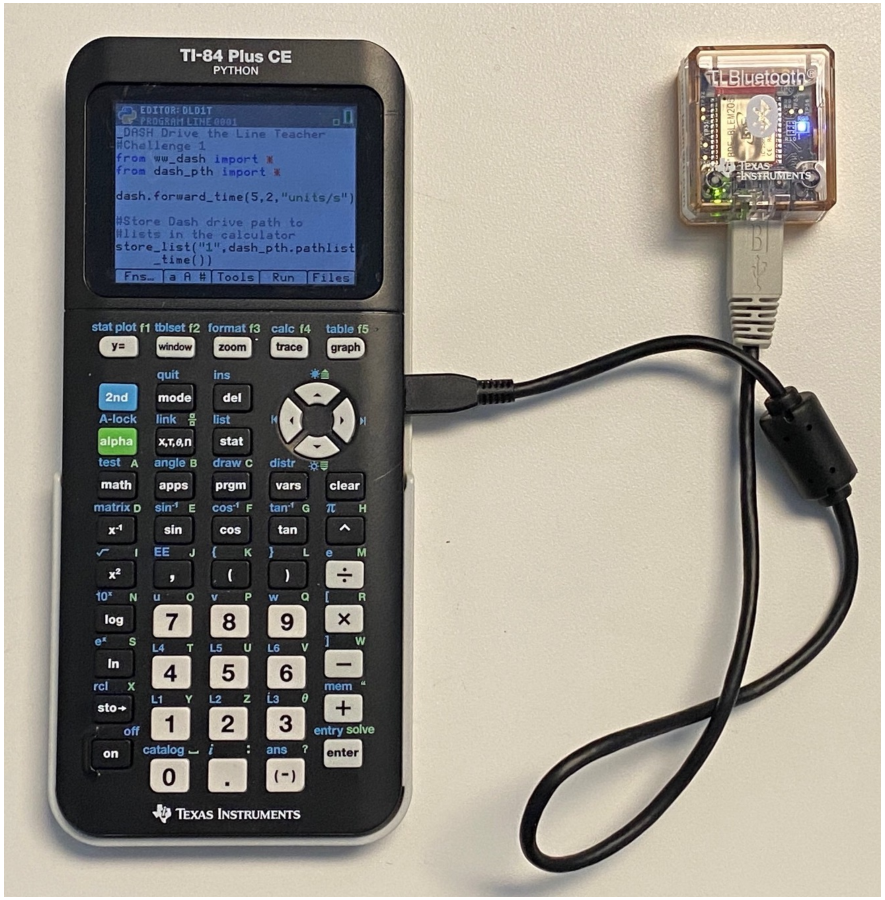
Texas Instruments

@ticalculators

Meet the Wonder Workshop™ Dash



Control the Dash using Python from the TI-84 Plus CE over Bluetooth Wireless



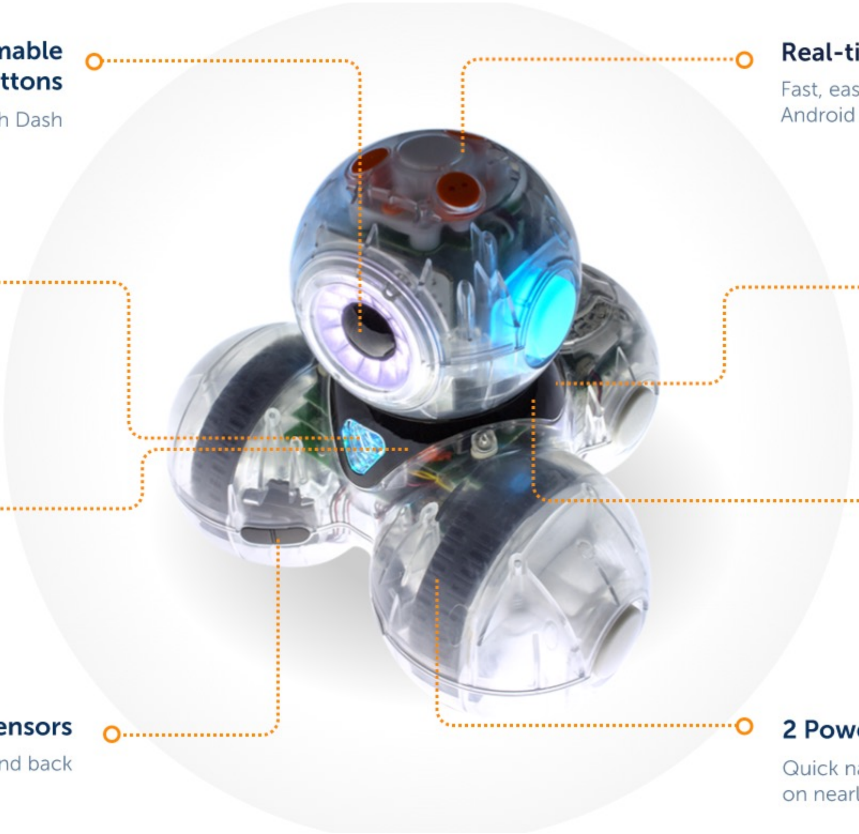
```
EDITOR: DLD1T
PROGRAM LINE 0001
_DASH Drive the Line Teacher
#Challenge 1
from ww_dash import *
from dash_pth import *

dash.forward_time(5,2,"units/s")

#Store Dash drive path to
#lists in the calculator
store_list("1",dash_pth.pathlist
    _time())
```

TI Bluetooth Adapter

Wonder Workshop™ Dash

TI-84 Plus CE Python

Meet the Wonder Workshop™ Dash



User Programmable LED's and Buttons

Customize your experience with Dash

Real-time Bluetooth

Fast, easy connections to Apple iOS, Android and Kindle mobile devices

IR Receivers & Transmitters

Enables Dash to find and interact with other robots

3x Microphones & Speakers

Real-time voice triangulation and personalized recording and playback

Potentiometers & Dual Motors

Supports head pan and tilt with accurate positioning

3 Processors & Sensor Fusion

Manages complex interactions among actuators & sensors - accelerometer, gyroscope and wheel encoders

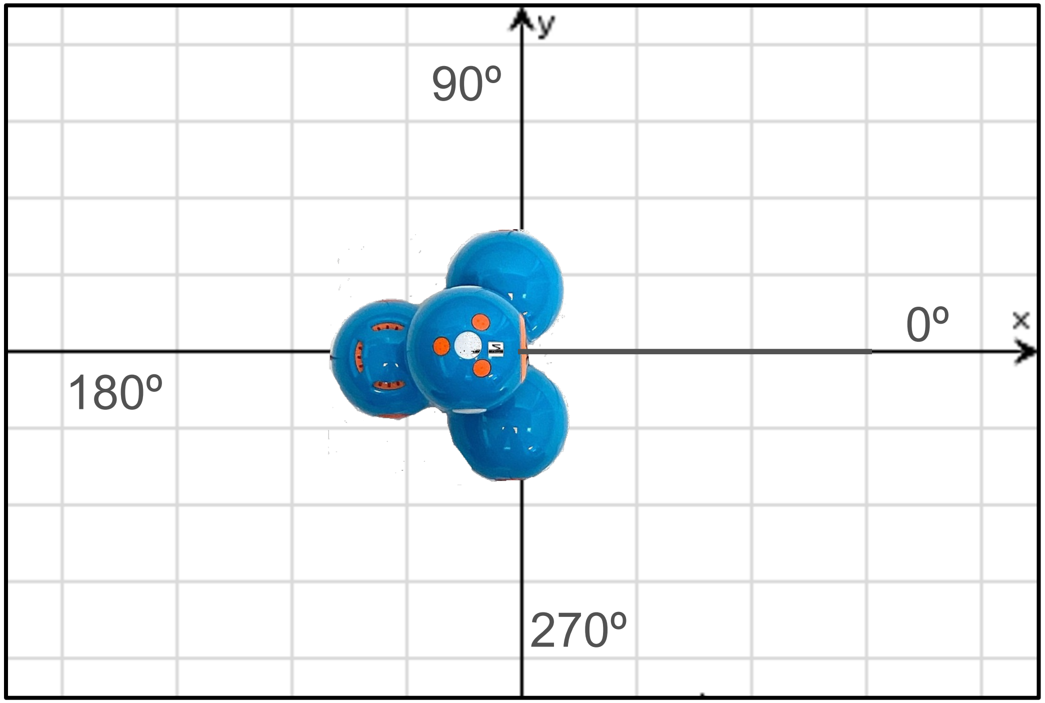
3 Proximity Sensors

Detects objects left, right, and back

2 Powered Wheels

Quick navigation and distance tracking on nearly any surface

Dash orientation and virtual grid

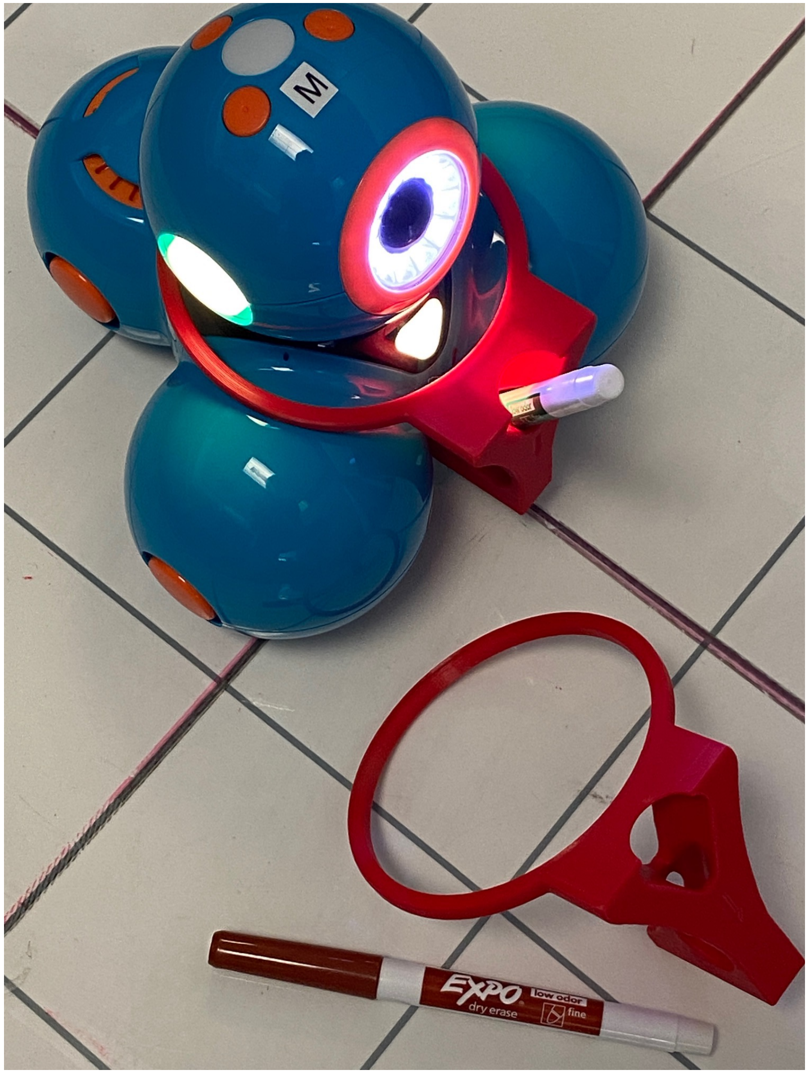


Dash programs set the initial position as the origin and the heading as 0 degrees measured from the x-axis.

Note: The Dash 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 Dash Drive menu. The virtual grid also applies to Path menu functions.

Draw with your Dash

Optional

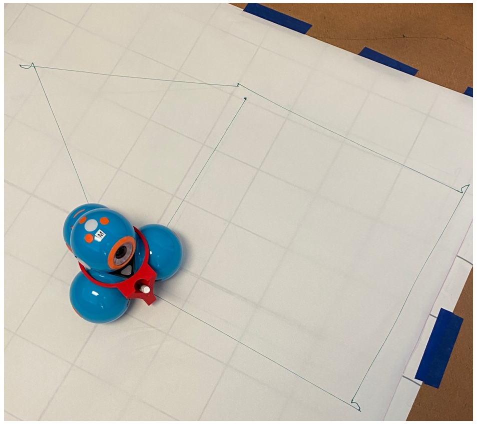


3D Print a snap-on attachment to hold Expo Fine dry erase markers for your Dash.

Step 1: Download .stl file at this link.

Step 2: 3D print following these recommendations.

1. Material PLA
2. Supports not recommended.
3. 20% infill
4. 3 to 4 shells
5. 0.2mm layer height



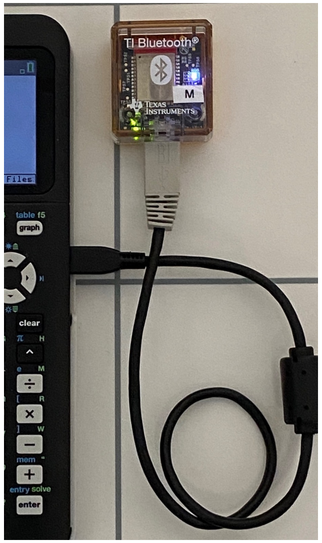
Setting up your calculator and TI Bluetooth adapter to run Dash

Find step-by-step directions in the Getting Started Guide at education.ti.com/dash

Follow the steps below to put the necessary files onto your calculator and to pair your TI Bluetooth Adapter with a Dash.

1 Download the ww_dash Python module and the SetDash App files to your calculator using the CE Connect computer application.

2 Plug the TI Bluetooth Adapter into your calculator.



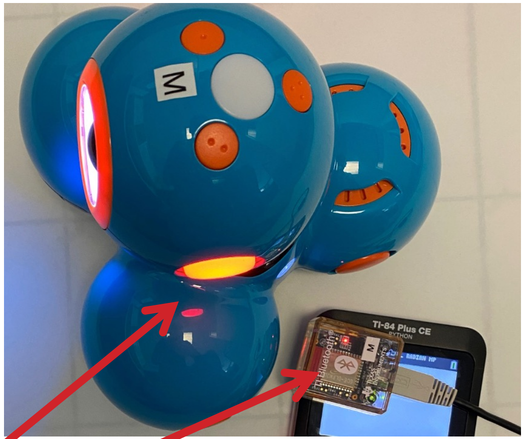
3 Turn on a Dash.
Use the SetDash App to search for and select a Dash to pair with the TI Bluetooth Adapter.

The Adapter “remembers” the Dash that it is paired with until you use the SetDash App to make a change.

Students can share a Dash by passing a paired Adapter to plug into their calculator.

Use stick-on letters or names to identify Dash/Adapter pairs. (”M” is used in the photo.)

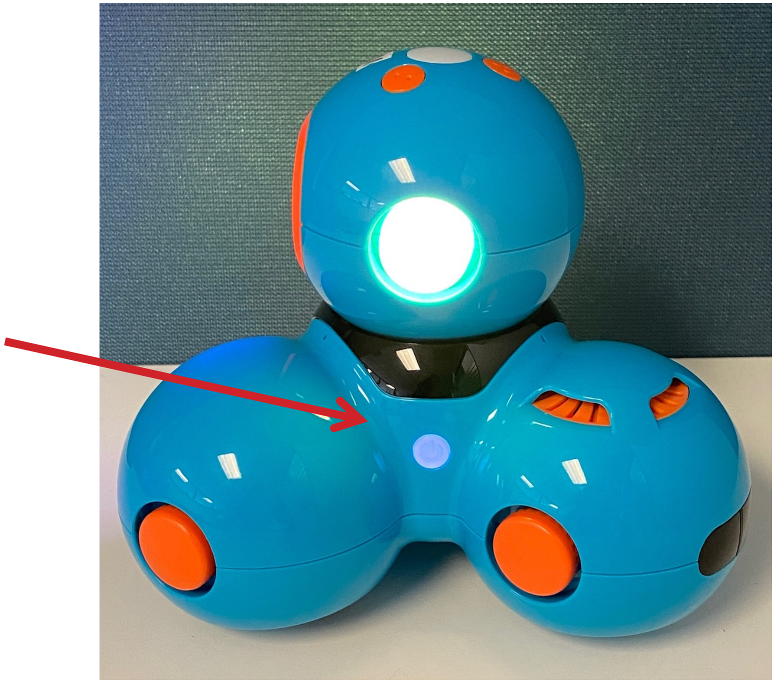
Note: The color of the adapter LED and the paired Dash LED’s match (Red in the Photo.). Use the SetDash App to control the Dash color.



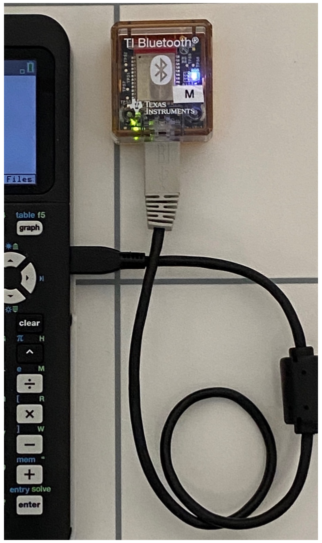
4 Quit the SetDash App. You are now ready to write and run Python programs to control the Dash.

Getting ready to run a Dash program

1 Make sure that your Dash is switched on.



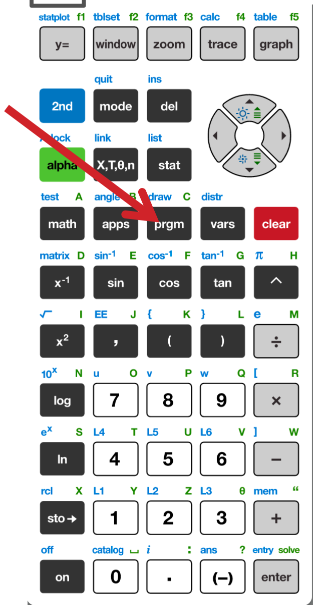
2 Plug the Bluetooth Adapter that is paired with the Dash into the calculator.



You are now ready to run Python programs that control the Dash.

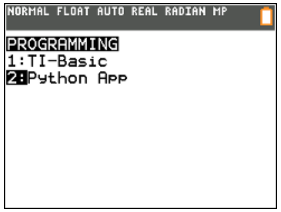
Creating a new Python Program

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.

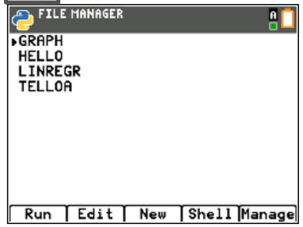
2



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

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

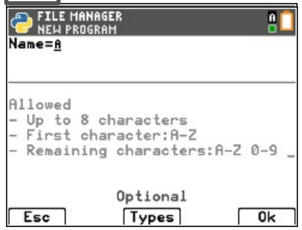
3



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

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

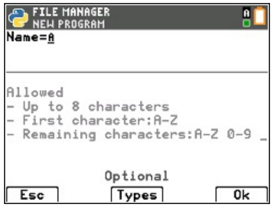
4



```
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 (zoom button)

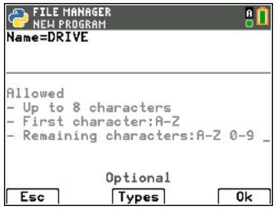
5



```
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.

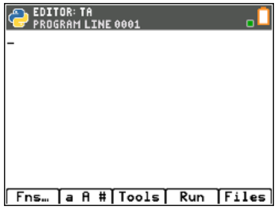
6



```
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



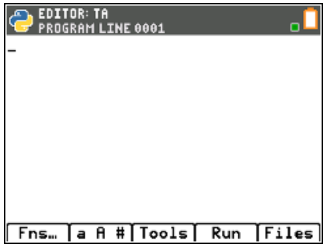
```
EDITOR: TA
PROGRAM LINE 0001
_






Fns…  a A #  Tools  Run  Files
```

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

Entering a Dash Program –

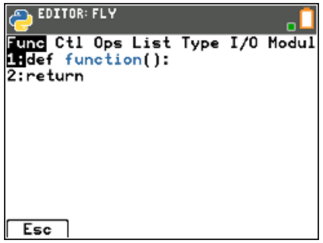
importing the Dash module

1



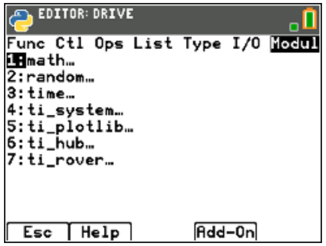
The Python program editor uses an insert cursor and a backspace delete.
Press **[Fns…]** softkey to see functions to use in your program.

2



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

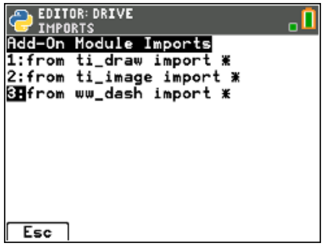
3



You will see a menu of installed modules available to use functions from.

You will need to add ww_dash to the installed module list.
Press **[Add-On]** softkey (trace button).

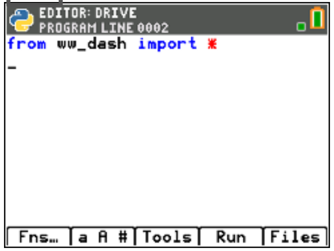
4



Select **from ww_dash import *** from the Add-On menu.

Use **down arrow**, if necessary, to move to the selection, then Press **[enter]**.

5



```
from ww_dash import *
```

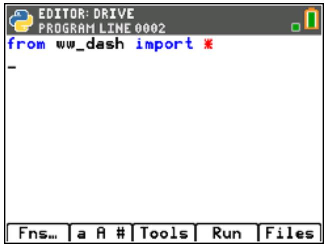
The ww_dash module import statement is pasted to your program.

The ww_dash import statement is required at the beginning of every Dash program.

This import statement brings in Dash functions to use in your program, sets Dash's initial position and sets up communication between the calculator and the Bluetooth Adapter.

Entering a Dash Program

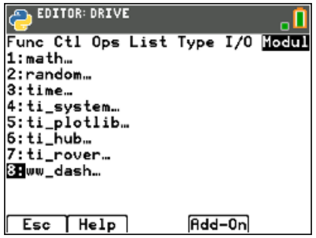
1



```
from ww_dash import *
_
```

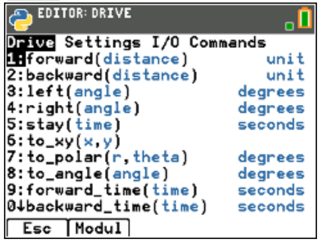
You are now ready to enter functions to control your Dash. Navigate to the Dash menus by pressing **[Fns…]** then **arrow** to the Modul menu.

2



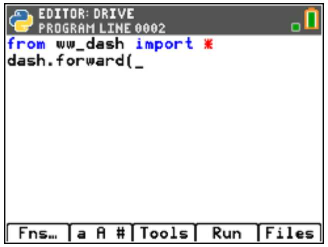
Then **select ww_dash…** to see options.

3


EDITOR: DRIVE
Drive Settings I/O Commands
1:forward(distance) unit
2:backward(distance) unit
3:left(angle) degrees
4:right(angle) degrees
5:stay(time) seconds
6:to_xy(x,y)
7:to_polar(r,theta) degrees
8:to_angle(angle) degrees
9:forward_time(time) seconds
0↓backward_time(time) seconds
Esc | Modul

You begin on the Drive menu. **Select** the **1:forward()** function.

4



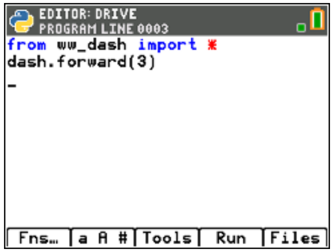
```
from ww_dash import *
dash.forward(_
```

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

A faster approach is to use **[2nd] [enter]** from any place on a line to complete the statement and move the cursor to the beginning of a blank line below.

Note: It is important that each statement begin on a new line.

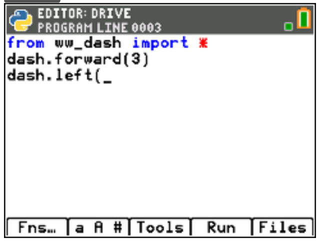
5



```
from ww_dash import *
dash.forward(3)
_
```

Navigate to the Drive menu again by press **[fns…]**, **left arrow**, select ww_dash**…**,**3:left()** to select the **left turn function**.

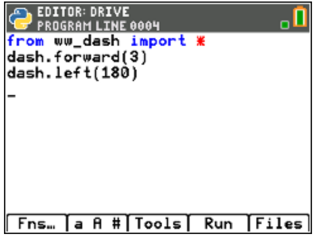
6



```
from ww_dash import *
dash.forward(3)
dash.left(_
```

Enter a value for the angle to turn in degrees. Press **[2nd] [enter]** to complete the statement and move to the beginning of new statement on the line below.

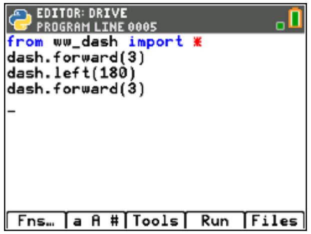
7



```
from ww_dash import *
dash.forward(3)
dash.left(180)
_
```

Navigate to the Drive menu again, then select **1:forward()**. After the function is pasted **enter the Dash units** to drive. Press **[2nd] [enter]** to complete the statement.

8

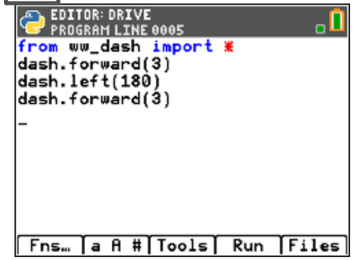


```
from ww_dash import *
dash.forward(3)
dash.left(180)
dash.forward(3)
_
```

You are now ready to run your Dash program.
.

Running a Dash Program

1



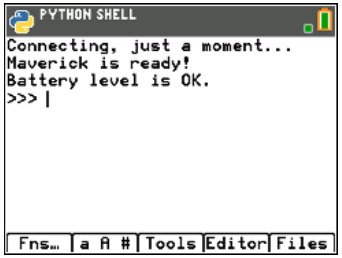
```
from ww_dash import *
dash.forward(3)
dash.left(180)
dash.forward(3)

_
```

You are now ready to run your program.

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

1. Make sure that Dash is turned ON.
2. Make sure that the calculator is connected to the Bluetooth Adapter.
3. Press **[Run]**.

2

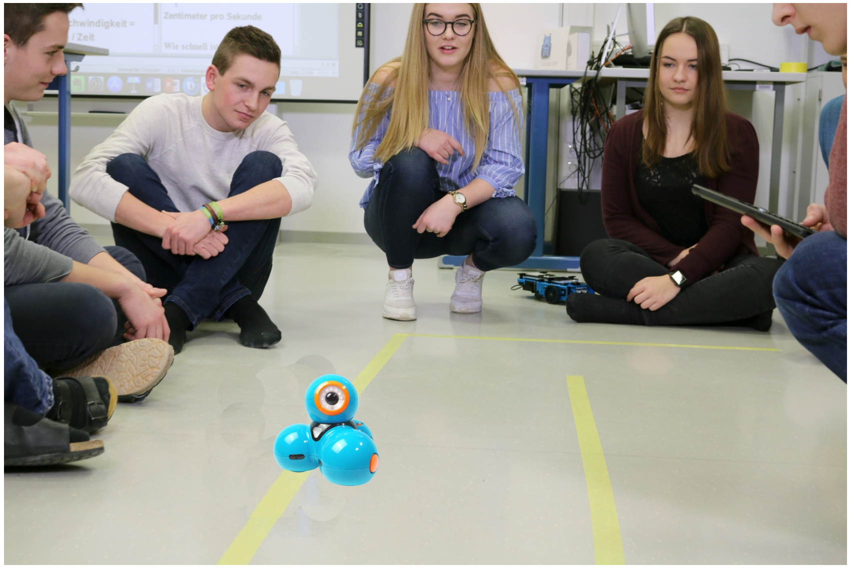


```
Connecting, just a moment...
Maverick is ready!
Battery level is OK.
>>> |
```

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

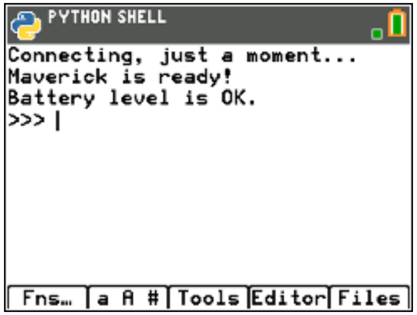
You can run the program again by pressing **[Tools]** and selecting **1:Rerun Last Program** from the menu.

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



Editing a Dash Program

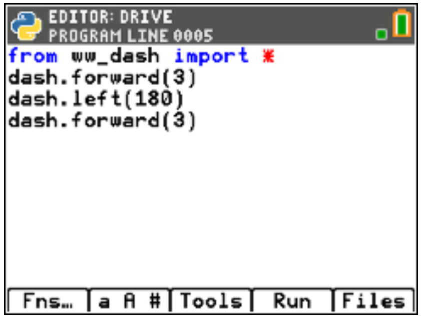
1



```
Connecting, just a moment...
Maverick is ready!
Battery level is OK.
>>> |
```

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

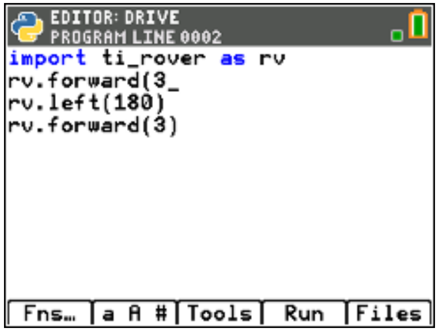
2



```
from ww_dash import *
dash.forward(3)
dash.left(180)
dash.forward(3)
```

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

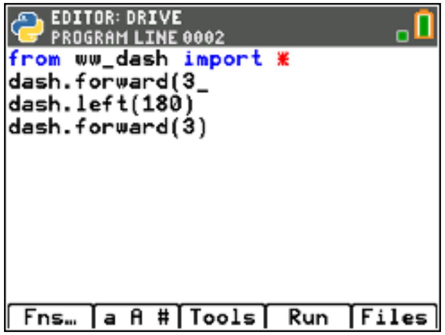
3



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

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

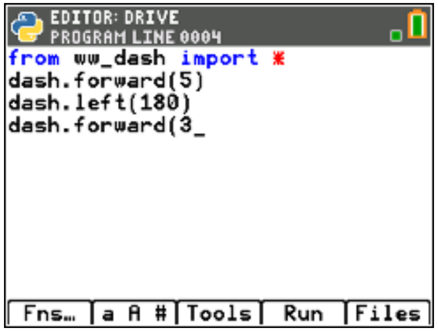
4



```
from ww_dash import *
dash.forward(3_
dash.left(180)
dash.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.

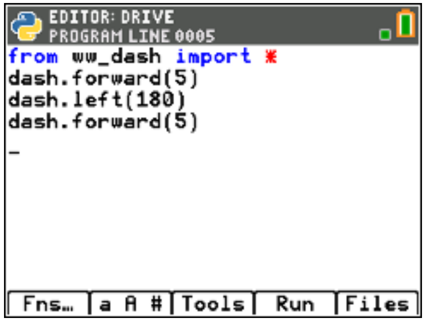
5



```
from ww_dash import *
dash.forward(5)
dash.left(180)
dash.forward(3_
```

Press **[del]** to backspace over the current distance value. **Type in a new value** for distance.
Press **[2nd] [enter]** to complete the statement.

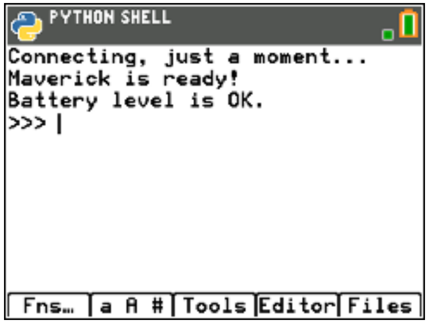
6



```
from ww_dash import *
dash.forward(5)
dash.left(180)
dash.forward(5)
_
```

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

7

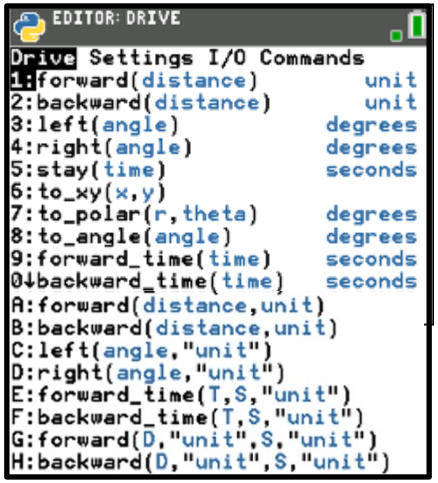


```
Connecting, just a moment...
Maverick is ready!
Battery level is OK.
>>> |
```

Texas Instruments

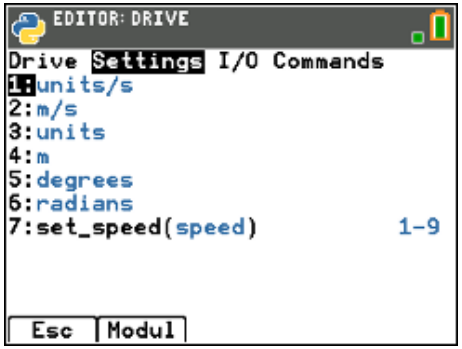
Dash Module Menus

Drive



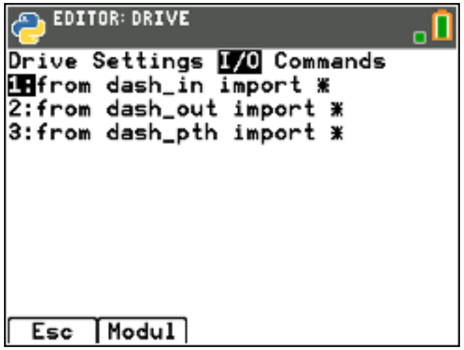
```
EDITOR: DRIVE
Drive Settings I/O Commands
1:forward(distance)        unit
2:backward(distance)       unit
3:left(angle)           degrees
4:right(angle)          degrees
5:stay(time)            seconds
6:to_xy(x,y)
7:to_polar(r,theta)     degrees
8:to_angle(angle)       degrees
9:forward_time(time)    seconds
0↓backward_time(time)   seconds
A:forward(distance,unit)
B:backward(distance,unit)
C:left(angle,"unit")
D:right(angle,"unit")
E:forward_time(T,S,"unit")
F:backward_time(T,S,"unit")
G:forward(D,"unit",S,"unit")
H:backward(D,"unit",S,"unit")
```

Settings



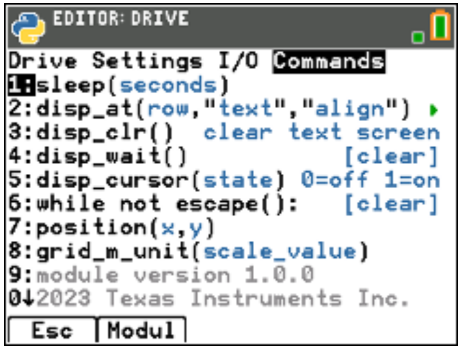
```
EDITOR: DRIVE
Drive Settings I/O Commands
1:units/s
2:m/s
3:units
4:m
5:degrees
6:radians
7:set_speed(speed)          1-9
 Esc  Modul
```

Input/Output (I/O)



```
EDITOR: DRIVE
Drive Settings I/O Commands
1:from dash_in import *
2:from dash_out import *
3:from dash_pth import *
 Esc  Modul
```

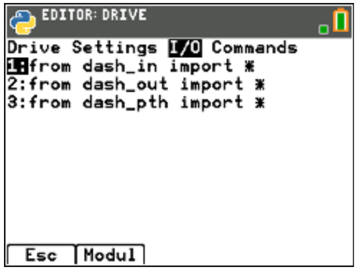
Commands



```
EDITOR: DRIVE
Drive Settings I/O Commands
1:sleep(seconds)
2:disp_at(row,"text","align") ▸
3:disp_clr()  clear text screen
4:disp_wait()              [clear]
5:disp_cursor(state) 0=off 1=on
6:while not escape():      [clear]
7:position(x,y)
8:grid_m_unit(scale_value)
9:module version 1.0.0
0↓2023 Texas Instruments Inc.
 Esc  Modul
```

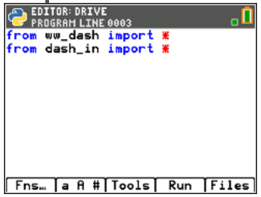
Dash Module Menus – I/O

Input/Output (I/O)

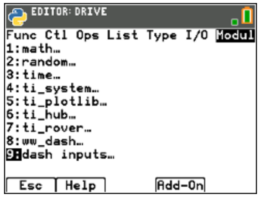


```
EDITOR: DRIVE
Drive Settings I/O Commands
1:from dash_in import *
2:from dash_out import *
3:from dash_pth import *
Esc | Modul
```

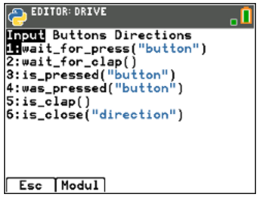
Inputs



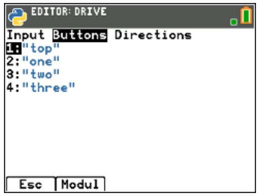
```
EDITOR: DRIVE
PROGRAM LINE 0003
from ww_dash import *
from dash_in import *
Fns… | a A # | Tools | Run | Files
```



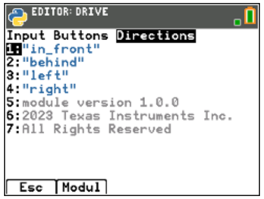
```
EDITOR: DRIVE
Func Ctl Ops List Type I/O Modul
1:math…
2:random…
3:time…
4:ti_system…
5:ti_plotlib…
6:ti_hub…
7:ti_rover…
8:ww_dash…
9:dash inputs…
Esc | Help | Add-On
```



```
EDITOR: DRIVE
Input Buttons Directions
1:wait_for_press("button")
2:wait_for_clap()
3:is_pressed("button")
4:was_pressed("button")
5:is_clap()
6:is_close("direction")
Esc | Modul
```

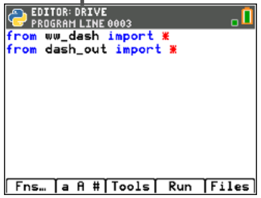


```
EDITOR: DRIVE
Input Buttons Directions
1:"top"
2:"one"
3:"two"
4:"three"
Esc | Modul
```

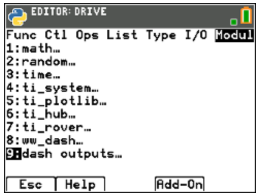


```
EDITOR: DRIVE
Input Buttons Directions
1:"in_front"
2:"behind"
3:"left"
4:"right"
5:module version 1.0.0
6:2023 Texas Instruments Inc.
7:All Rights Reserved
Esc | Modul
```

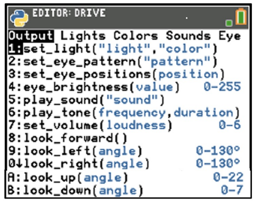
Outputs



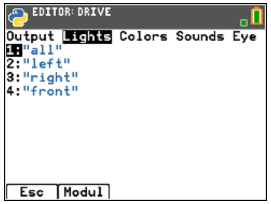
```
EDITOR: DRIVE
PROGRAM LINE 0003
from ww_dash import *
from dash_out import *
Fns… | a A # | Tools | Run | Files
```



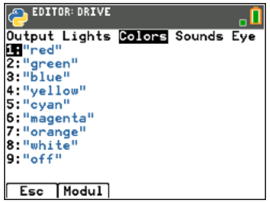
```
EDITOR: DRIVE
Func Ctl Ops List Type I/O Modul
1:math…
2:random…
3:time…
4:ti_system…
5:ti_plotlib…
6:ti_hub…
7:ti_rover…
8:ww_dash…
9:dash outputs…
Esc | Help | Add-On
```



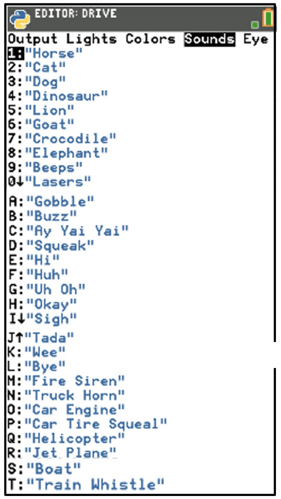
```
EDITOR: DRIVE
Output Lights Colors Sounds Eye
1:set_light("light","color")
2:set_eye_pattern("pattern")
3:set_eye_positions(position)
4:eye_brightness(value)      0-255
5:play_sound("sound")
6:play_tone(frequency,duration)
7:set_volume(loudness)         0-6
8:look_forward()
9:look_left(angle)         0-130°
0↓look_right(angle)        0-130°
A:look_up(angle)             0-22
B:look_down(angle)            0-7
```



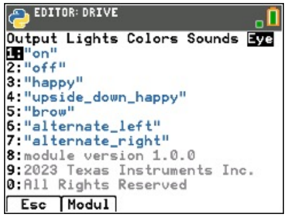
```
EDITOR: DRIVE
Output Lights Colors Sounds Eye
1:"all"
2:"left"
3:"right"
4:"front"
Esc | Modul
```



```
EDITOR: DRIVE
Output Lights Colors Sounds Eye
1:"red"
2:"green"
3:"blue"
4:"yellow"
5:"cyan"
6:"magenta"
7:"orange"
8:"white"
9:"off"
Esc | Modul
```

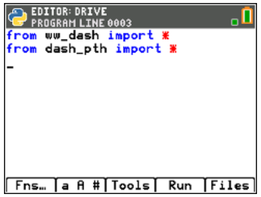


```
EDITOR: DRIVE
Output Lights Colors Sounds Eye
1:"Horse"
2:"Cat"
3:"Dog"
4:"Dinosaur"
5:"Lion"
6:"Goat"
7:"Crocodile"
8:"Elephant"
9:"Beeps"
0↓"Lasers"
A:"Gobble"
B:"Buzz"
C:"Ay Yai Yai"
D:"Squeak"
E:"Hi"
F:"Huh"
G:"Uh Oh"
H:"Okay"
I↓"Sigh"
J↑"Tada"
K:"Wee"
L:"Bye"
M:"Fire Siren"
N:"Truck Horn"
O:"Car Engine"
P:"Car Tire Squeal"
Q:"Helicopter"
R:"Jet Plane"
S:"Boat"
T:"Train Whistle"
```

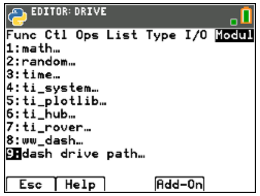


```
EDITOR: DRIVE
Output Lights Colors Sounds Eye
1:"on"
2:"off"
3:"happy"
4:"upside_down_happy"
5:"brow"
6:"alternate_left"
7:"alternate_right"
8:module version 1.0.0
9:2023 Texas Instruments Inc.
0:All Rights Reserved
Esc | Modul
```

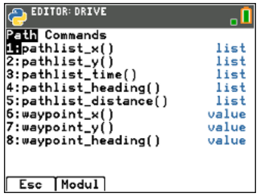
Path



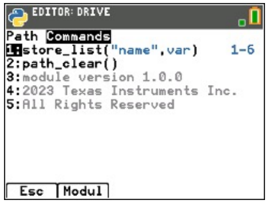
```
EDITOR: DRIVE
PROGRAM LINE 0003
from ww_dash import *
from dash_pth import *
-
Fns… | a A # | Tools | Run | Files
```



```
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…
8:ww_dash…
9:dash drive path…
Esc | Help | Add-On
```



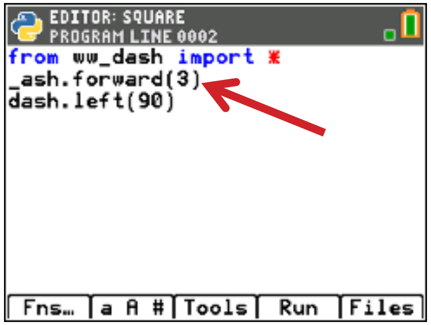
```
EDITOR: DRIVE
Path Commands
1:pathlist_x()               list
2:pathlist_y()               list
3:pathlist_time()            list
4:pathlist_heading()         list
5:pathlist_distance()        list
6:waypoint_x()              value
7:waypoint_y()              value
8:waypoint_heading()        value
Esc | Modul
```



```
EDITOR: DRIVE
Path Commands
1:store_list("name",var)     1-6
2:path_clear()
3:module version 1.0.0
4:2023 Texas Instruments Inc.
5:All Rights Reserved
Esc | Modul
```

Copying and Pasting a Line of Code

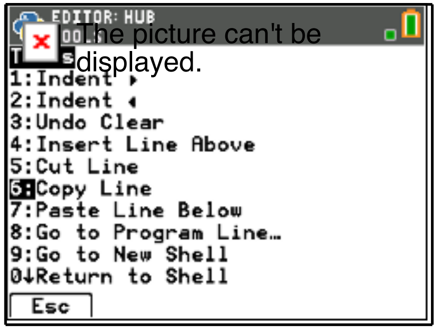
1



```
from ww_dash import *
_ash.forward(3)
dash.left(90)
```

Use **arrow keys** to move the cursor to a position anywhere on the line that you would like to copy.

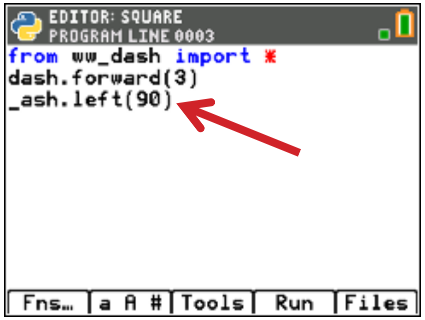
2



Press **[Tools]** then select **6:Copy Line** from the menu.

After you select you will be returned to the editor.

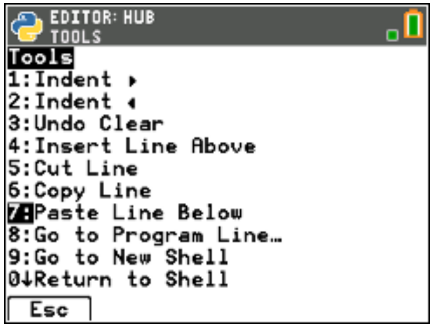
3



```
from ww_dash import *
dash.forward(3)
_ash.left(90)
```

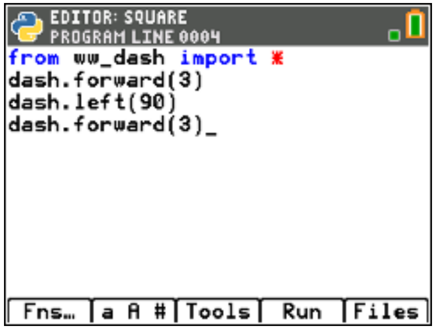
Use **arrow keys** to move the cursor to any location on the line above where you would like to insert the copied line.

4



Press **[Tools]** then select **7:Paste Line Below** from the menu. The copied line will be pasted.

5

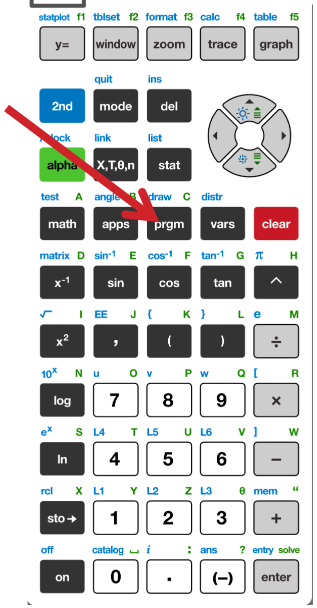


```
from ww_dash import *
dash.forward(3)
dash.left(90)
dash.forward(3)_
```

You can paste again by returning to the **[Tools]** menu and selecting **7:Paste Line Below.**

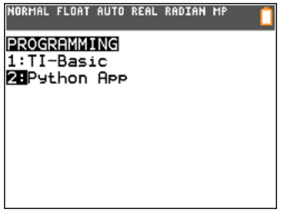
Opening an existing Python Program File

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.

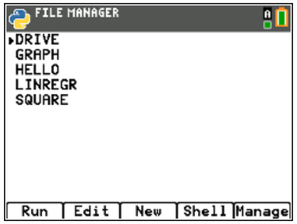
2



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

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

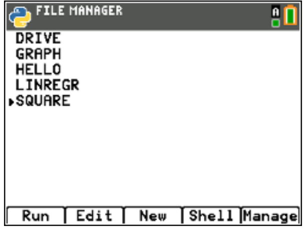
3



```
FILE MANAGER
▸DRIVE
 GRAPH
 HELLO
 LINREGR
 SQUARE
```

To edit an existing program, use the **Up and Down Arrow keys** to select a program.

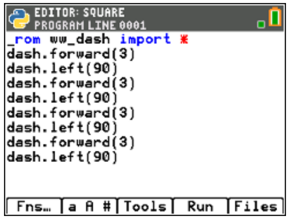
4



```
FILE MANAGER
 DRIVE
 GRAPH
 HELLO
 LINREGR
▸SQUARE
```

Press **[Edit]** to open with Python Editor with the selected program.

5

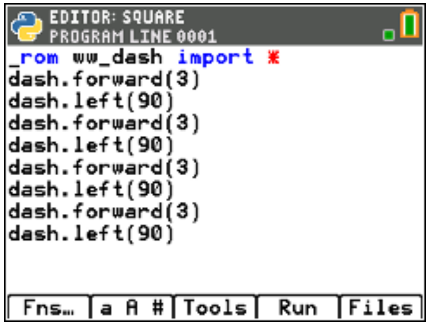


```
EDITOR: SQUARE
PROGRAM LINE 0001
_rom ww_dash import *
dash.forward(3)
dash.left(90)
dash.forward(3)
dash.left(90)
dash.forward(3)
dash.left(90)
dash.forward(3)
dash.left(90)
```

You can now make changes to the program or run the program.

Copying/Replicating a Python Program File

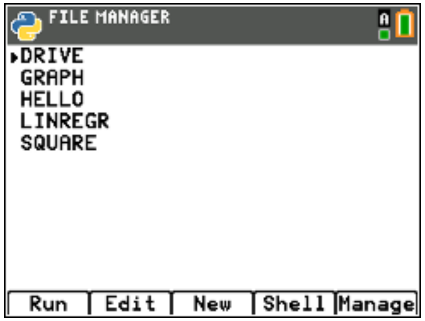
1



```
EDITOR: SQUARE
PROGRAM LINE 0001
_rom ww_dash import *
dash.forward(3)
dash.left(90)
dash.forward(3)
dash.left(90)
dash.forward(3)
dash.left(90)
dash.forward(3)
dash.left(90)
```

Press **[Files]** to return to the file management screen.

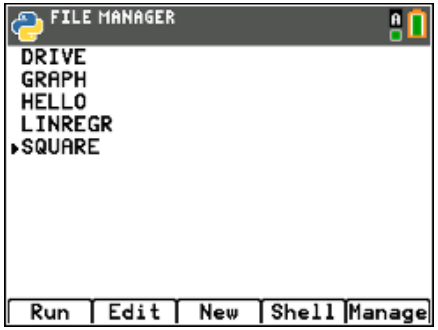
2



```
FILE MANAGER
▸DRIVE
 GRAPH
 HELLO
 LINREGR
 SQUARE
```

Use the **Up and Down Arrow keys** to select a program.

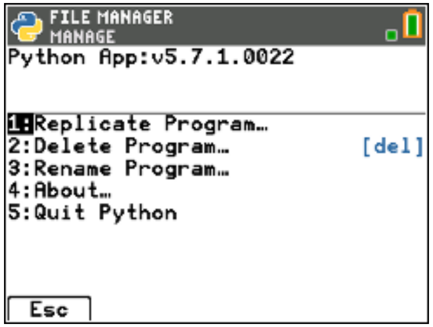
3



```
FILE MANAGER
 DRIVE
 GRAPH
 HELLO
 LINREGR
▸SQUARE
```

Press **[Manage]** to see file options.

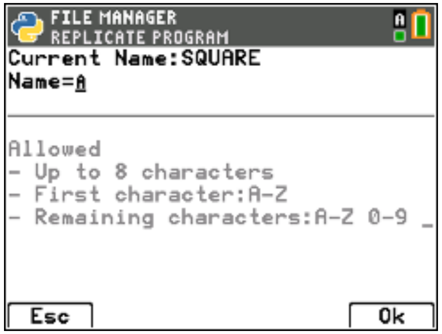
4



```
FILE MANAGER
MANAGE
Python App:v5.7.1.0022

1:Replicate Program…
2:Delete Program…          [del]
3:Rename Program…
4:About…
5:Quit Python
```

Select **1:Replicate Program** to receive a prompt.

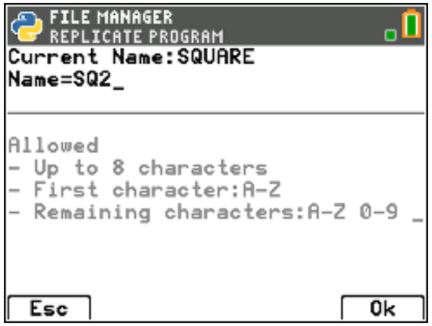
5



```
FILE MANAGER
REPLICATE PROGRAM
Current Name:SQUARE
Name=A

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

Type in the name of the new program using the green alpha key labels.

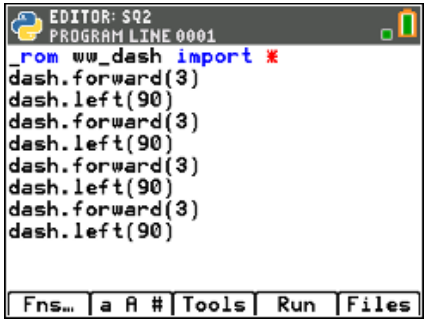
6



```
FILE MANAGER
REPLICATE PROGRAM
Current Name:SQUARE
Name=SQ2_

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

To use a number in the name, exit alpha mode by pressing **[2nd] [alpha]** then a **number key**.
Press **[Ok]** to finish the dialogue.

7



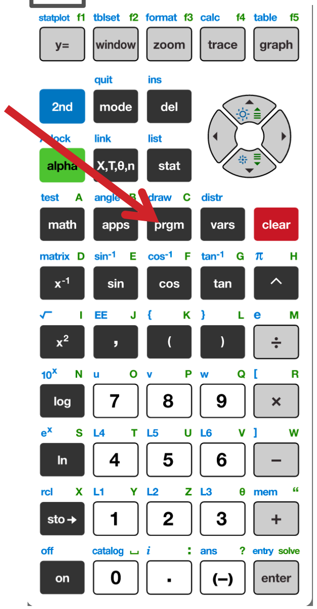
```
EDITOR: SQ2
PROGRAM LINE 0001
_rom ww_dash import *
dash.forward(3)
dash.left(90)
dash.forward(3)
dash.left(90)
dash.forward(3)
dash.left(90)
dash.forward(3)
dash.left(90)
```

You are now in the editor ready to make changes or to run the new program.

Creating a new Dash Python Program using WW Dash Helpers…

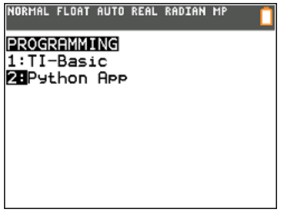
Use the Dash Helper menu when creating new programs to automatically paste import module statements and the necessary functions for different types of applications.

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

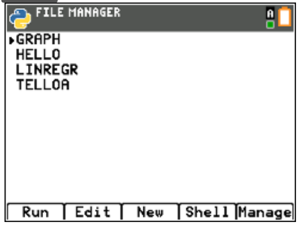
Press the **[prgm]** key to create, edit and execute TI-Python programs.

2



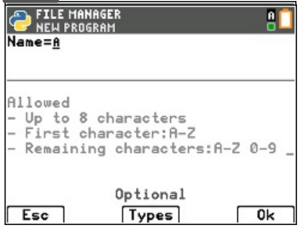
Press **down arrow** **[enter]** or Press **[2]** to select 2: Python App

3



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

4

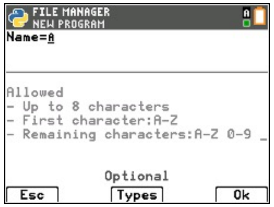


Name=A

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

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

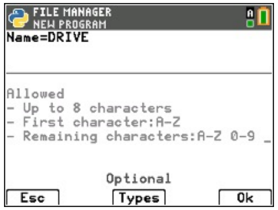
5



Name=A

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

6



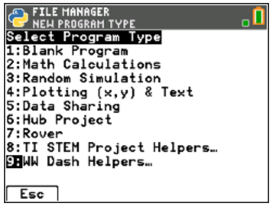
Name=DRIVE

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

Type your program name.

Then press the **[Types]** softkey to see options.

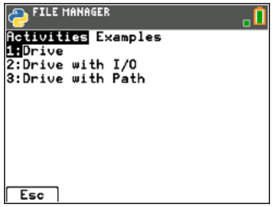
7



select 9: WW Dash Helpers… from the menu.

Dash Helpers are optional activities and examples that will paste the necessary module import statements

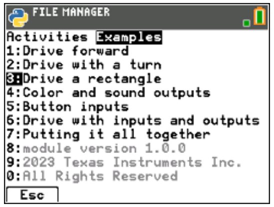
8



You have options to choose from a Menu of Activities that will paste the necessary module import statements and functions to begin your program.

Right arrow to see a list of example programs.

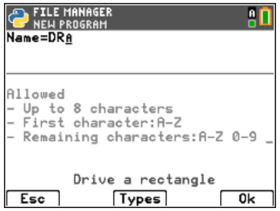
9



Selecting from this menu will paste the necessary statements for different programs that you may be of interest.

Press **[3]** to select 3:Drive a rectangle.

10



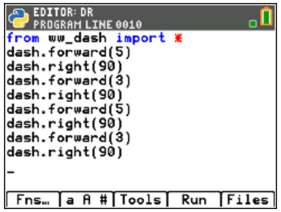
Name=DRA

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

Your Example selection is displayed on the screen just above the [Types] softkey label.

Press **[Ok]** to create the program using the Drive the Rectangle example.

11

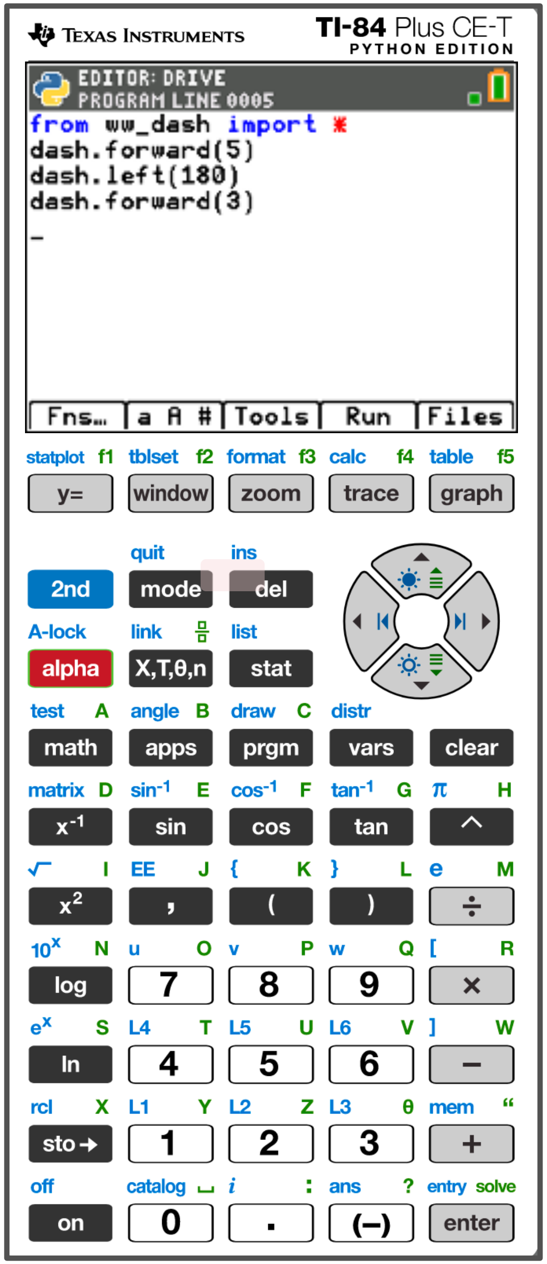


```
from ww_dash import *
dash.forward(5)
dash.right(90)
dash.forward(3)
dash.right(90)
dash.forward(5)
dash.right(90)
dash.forward(3)
dash.right(90)
```

The program is ready to edit or run.

Entry and Edit Tips

- Use **number key shortcuts** or **arrow keys** and **[enter]** to select from menus
- Use **arrow keys** to move the cursor around the screen.
- Use **[alpha] repeatedly** to cycle from numeric, to lower case alpha to upper case alpha entry mode. The cursor indicates the current mode.
- Use **[2nd] [A-lock]** to lock to alpha entry or to return to numeric entry.
- Use **[Fns…] softkey** to bring up Python function menus, including the **Modul (modules)** menu.
- Use **[clear]** or **[Esc] softkey** to back out of a menu.
- Use **[del]** as a destructive backspace
- Use **[2nd] [enter]** from any place on a line to complete the statement and move the cursor to the beginning of a blank line below.
- Use **[Tools] softkey** menu to undo a clear and to copy, cut, paste and more.
- Use **[Editor] softkey** to return to the editor from the Shell.
- Use **[2nd] [quit]** to leave the Python app and return to the calculator.



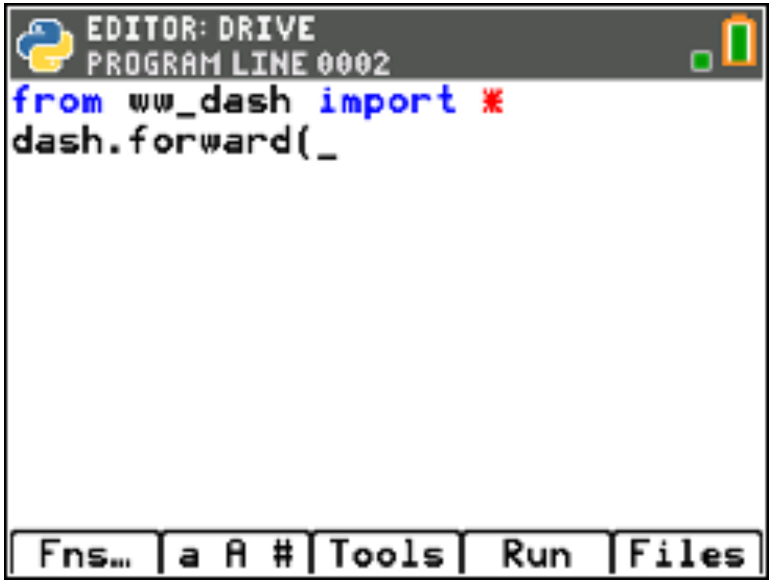
EDITOR: DRIVE
PROGRAM LINE 0005

```
from ww_dash import *
dash.forward(5)
dash.left(180)
dash.forward(3)
```

Texas Instruments
TI-84 Plus CE-T
PYTHON EDITION
Fns… | a A # | Tools | Run | Files
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

MAKE IT MOVE!

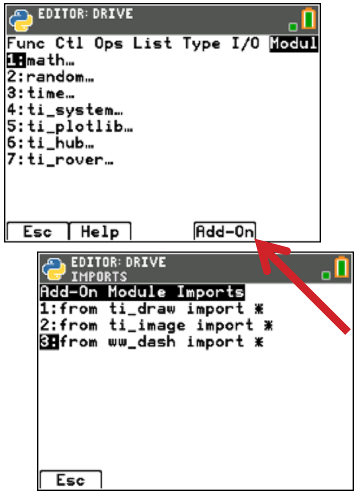
New Program:



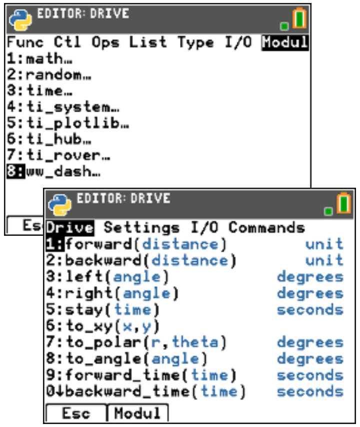
```
from ww_dash import *
dash.forward(_
```

Task: Discover how far Dash drives per unit.

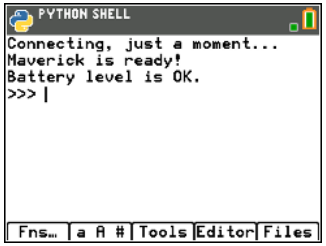
Use differing values (1-20) to determine what 1 Dash unit is



Press **[Fns…], left arrow,** then press the **[Add-On]** softkey to import the ww_dash Python module.



Press **[Fns…], left arrow,** then **select 8: ww_dash…** from the menu. Then select **1:forward()** function.



```
Connecting, just a moment...
Maverick is ready!
Battery level is OK.
>>> |
```

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

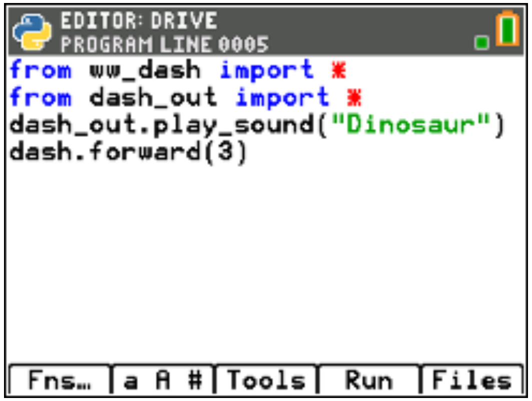
From the Python shell, press **[Editor]** to move from the shell to the Python editor.

Have Dash make a sound

Task: Have your Dash play a sound while driving.

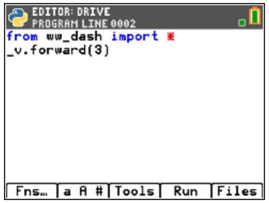
Challenge Task: Have your Dash make a "Fire Siren" sound.

Edit Program:

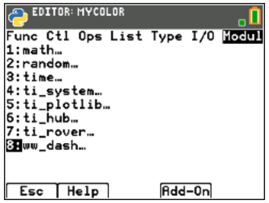


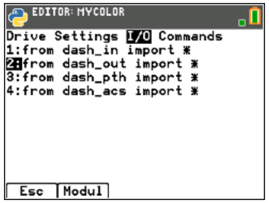
```
from ww_dash import *
from dash_out import *
dash_out.play_sound("Dinosaur")
dash.forward(3)
```

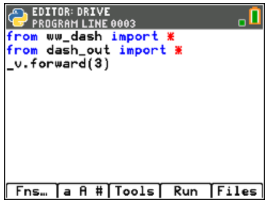
Import the dash_out Python module, which includes sound capabilities. Move your cursor to the beginning of the row with the forward function. Press **[Fns…], left arrow,** then **8:ww_dash…** then **right arrow** to the **I/O menu** and select **2:from dash_out import ***



```
from ww_dash import *
_v.forward(3)
```

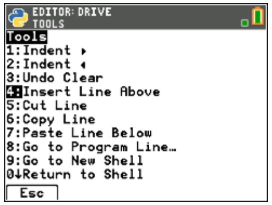


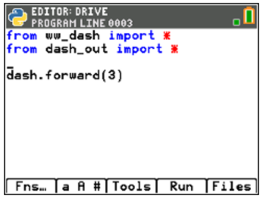




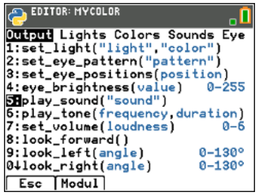
```
from ww_dash import *
from dash_out import *
_v.forward(3)
```

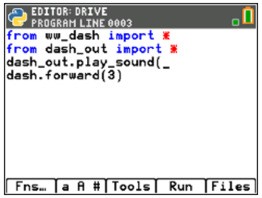
Add the play_sound function to the program**.**
Insert a blank row: **Press [Tools] 4:Insert Line Above].**
Then Press **[Fns…], left arrow,** then **9:dash_out…**for the Dash output menus.
Select **5:play_sound( )**





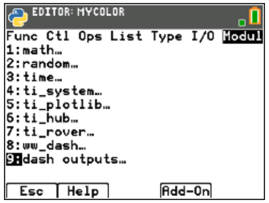
```
from ww_dash import *
from dash_out import *

dash.forward(3)
```

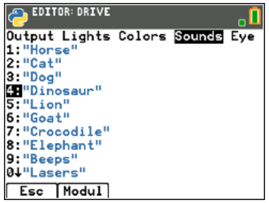

EDITOR: MYCOLOR
Output Lights Colors Sounds Eye
1:set_light("light","color")
2:set_eye_pattern("pattern")
3:set_eye_positions(position)
4:eye_brightness(value) 0-255
5:play_sound("sound")
6:play_tone(frequency,duration)
7:set_volume(loudness) 0-6
8:look_forward()
9:look_left(angle) 0-130°
0↓look_right(angle) 0-130°
Esc | Modul

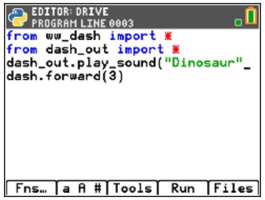


```
from ww_dash import *
from dash_out import *
dash_out.play_sound(_
dash.forward(3)
```

Select a sound to play.
Then Press **[Fns…], left arrow,** then **9:dash_out…** Then **right arrow** to the Sounds menu. Use the **up** and **down arrow keys** and **[enter]** or number and letter keys to choose your sound.
Press **[Run]** to run your program.







```
from ww_dash import *
from dash_out import *
dash_out.play_sound("Dinosaur"_
dash.forward(3)
```

Texas Instruments

Task: Set the color output of the Red, Green, Blue (RGB) LED.

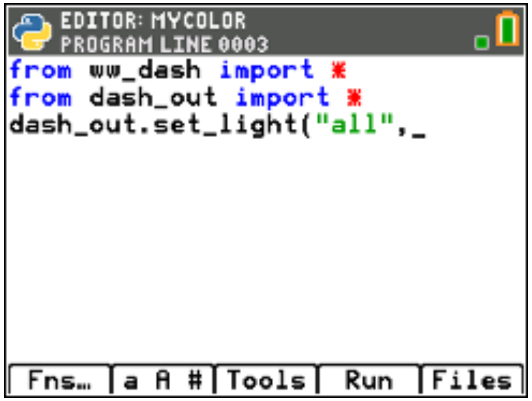
Each color takes a value of (0-255).

Challenge Tasks:

Try to make Yellow or Cyan or Magenta.

Extra Challenge: Make your own color and give the color a fun name.

New Program:

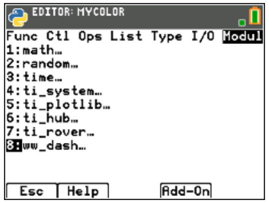


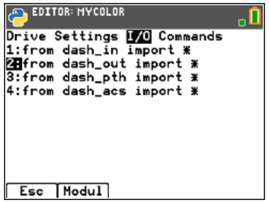
```
from ww_dash import *
from dash_out import *
dash_out.set_light("all",_
```

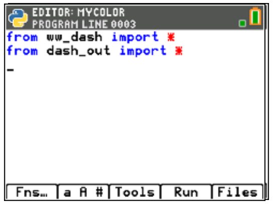
Set the color

Import the ww_dash Python module. Press **[Fns…], left arrow,** then press the **[Add-On]** softkey.

Import the dash_out Python module. Press **[Fns…], left arrow,** then **8:ww_dash…** then **right arrow** to the **I/O menu** and select **2:from dash_out import ***

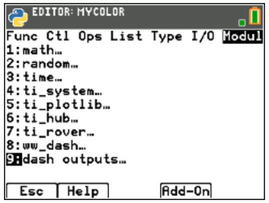


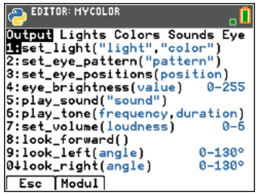


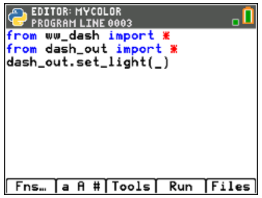


```
from ww_dash import *
from dash_out import *
_
```

Add the set_light function to the program. Press **[Fns…], left arrow,** then **9:dash_out…**for the Dash output menus. Select **1:set_light( )**



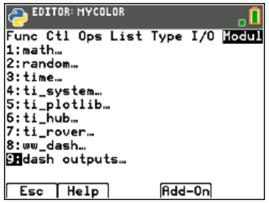

EDITOR: MYCOLOR
Output Lights Colors Sounds Eye
1:set_light("light","color")
2:set_eye_pattern("pattern")
3:set_eye_positions(position)
4:eye_brightness(value) 0-255
5:play_sound("sound")
6:play_tone(frequency,duration)
7:set_volume(loudness) 0-6
8:look_forward()
9:look_left(angle) 0-130°
0↓look_right(angle) 0-130°
Esc | Modul

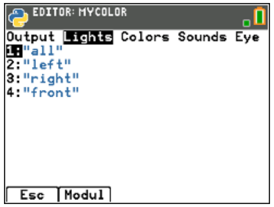


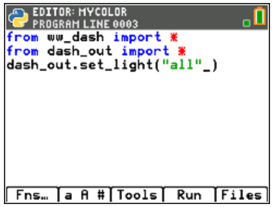
```
from ww_dash import *
from dash_out import *
dash_out.set_light(_)
```

Select which Dash lights to change and set the color of the lights.
Press **[Fns…], left arrow,** then **9:dash_out… Right arrow** to the Lights Menu. Select **1:"all"**.

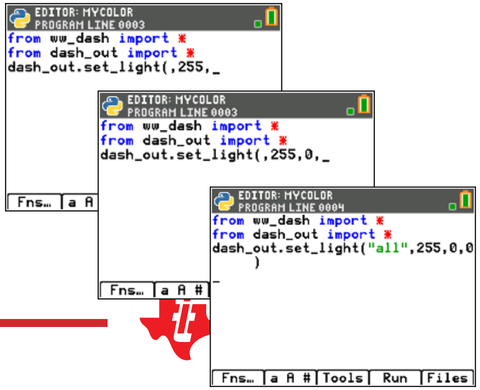
Then **right arrow** past the comma to begin entering the Red, Green and Blue values to create your color. Enter a value for the Red component of the color followed by a comma, then enter Green followed by a comma, then enter Blue followed by **[2nd] [enter]** to complete the statement.







```
from ww_dash import *
from dash_out import *
dash_out.set_light("all"_)
```



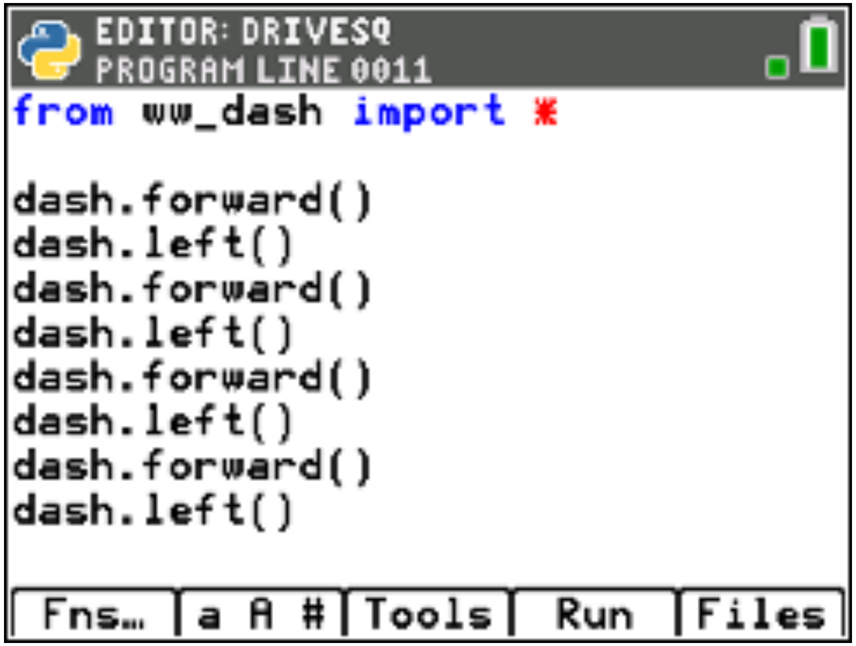
```
from ww_dash import *
from dash_out import *
dash_out.set_light(,255,_
```

```
from ww_dash import *
from dash_out import *
dash_out.set_light(,255,0,_
```

```
from ww_dash import *
from dash_out import *
dash_out.set_light("all",255,0,0
)
_
```

Explore angles

New Program:



```
from ww_dash import *

dash.forward()
dash.left()
dash.forward()
dash.left()
dash.forward()
dash.left()
dash.forward()
dash.left()
```

Press **[Fns…], left arrow,** then press the **[Add-On]** softkey to import the ww_dash Python module.

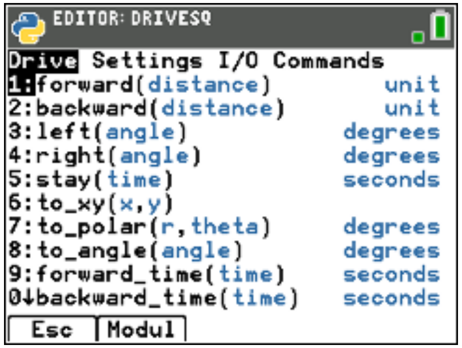
Press **[Fns…], left arrow,** then **8:ww_dash…** for the Dash menus.

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

Task: Drive a square.

Challenge Task: Try to drive an equilateral triangle.

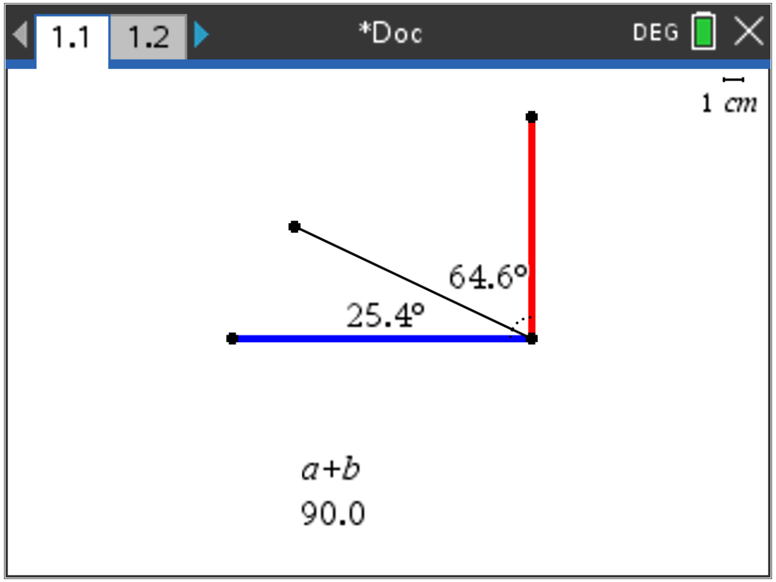
See the inputs for the most common drive functions below.



```
1:forward(distance)          unit
2:backward(distance)         unit
3:left(angle)             degrees
4:right(angle)            degrees
5:stay(time)              seconds
6:to_xy(x,y)
7:to_polar(r,theta)       degrees
8:to_angle(angle)         degrees
9:forward_time(time)      seconds
0↓backward_time(time)     seconds
```

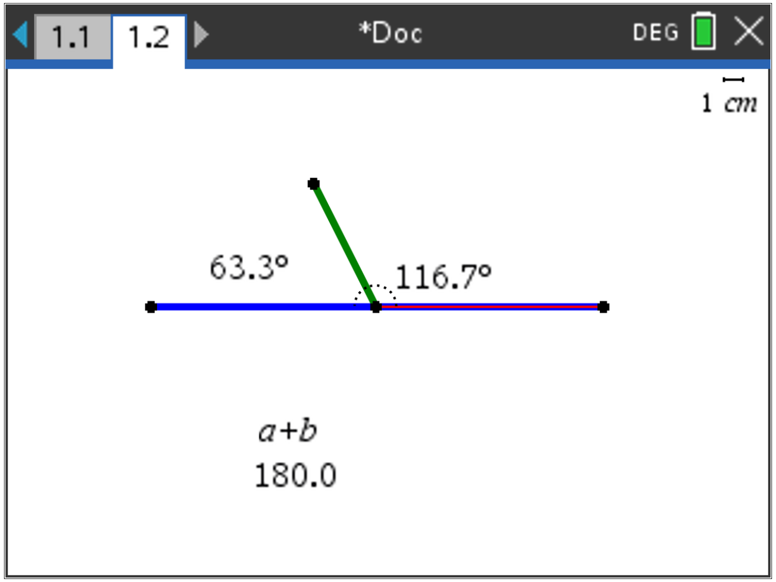
Quick Math Reminders

- Complementary Angles:
  - Sum to 90 degrees



a+b
90.0

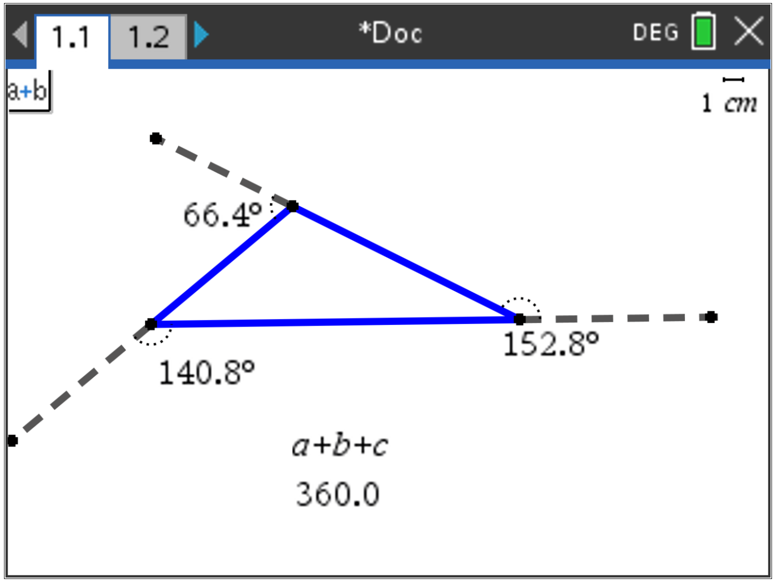
- Supplementary Angles:
  - Sum to 180 degrees

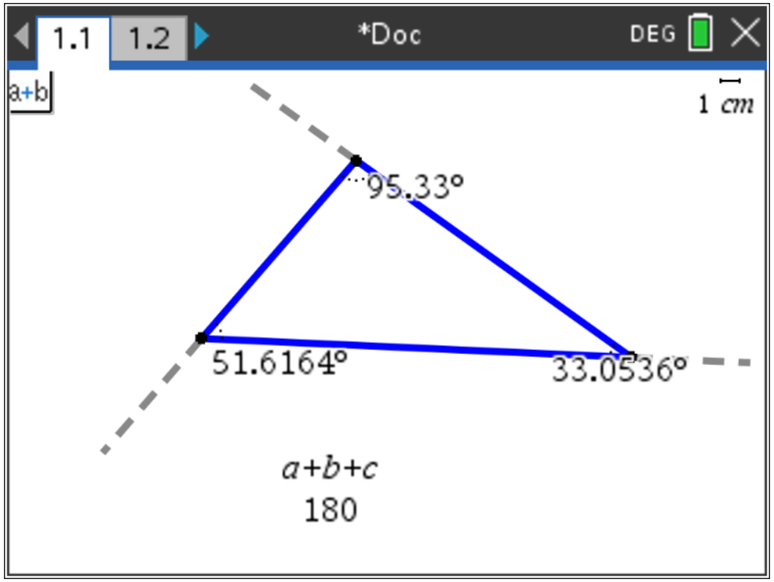


a+b
180.0

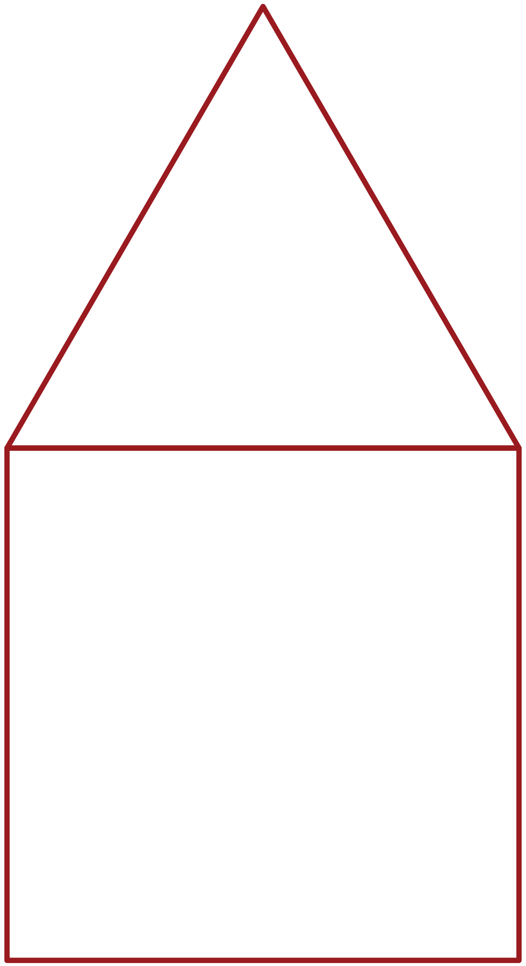
Quick Math Reminders

- » Exterior angles:
- » Interior Angles:





Logic Challenge



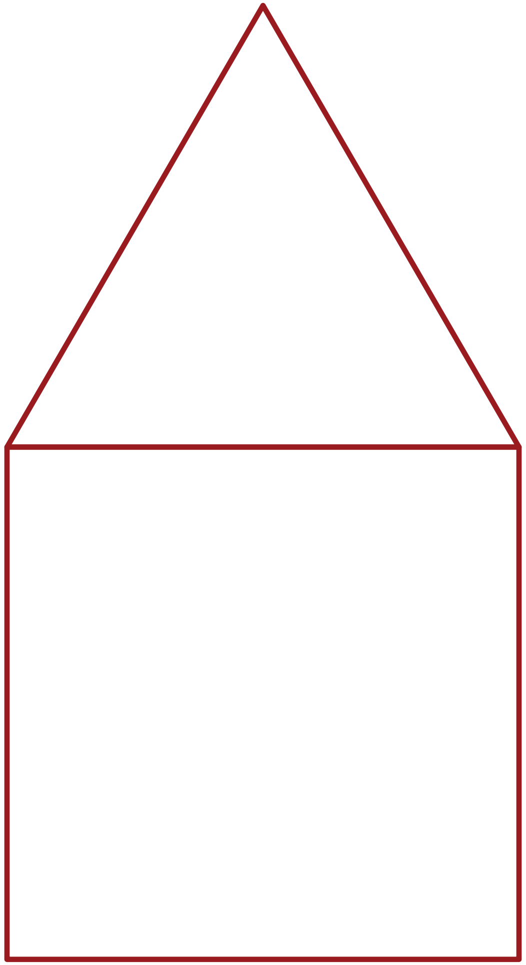
Task: Draw the figure shown large enough for Dash to drive.

Note: Try side lengths of ~ .4 Meters

Write a program to have your Dash drive the figure without crossing any lines or going back over a line.

Marker version

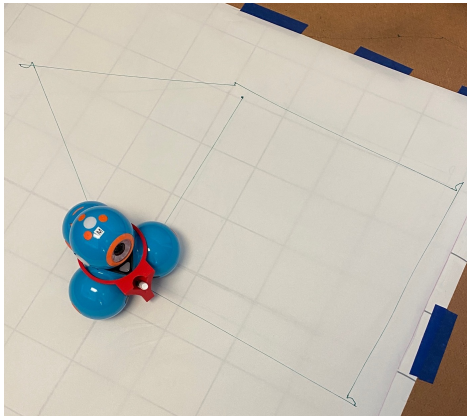
Logic Challenge



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

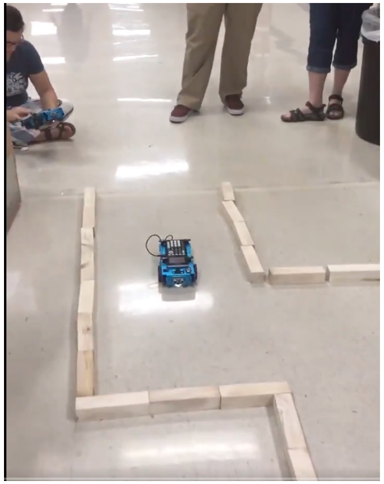
Note: Try side lengths of ~ .4 Meters

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

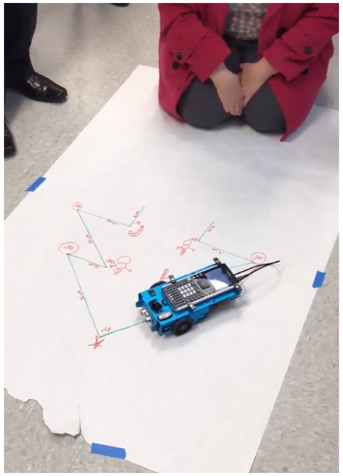


Note: For more information about marker holder 3D Print file see the earlier slide in this deck.

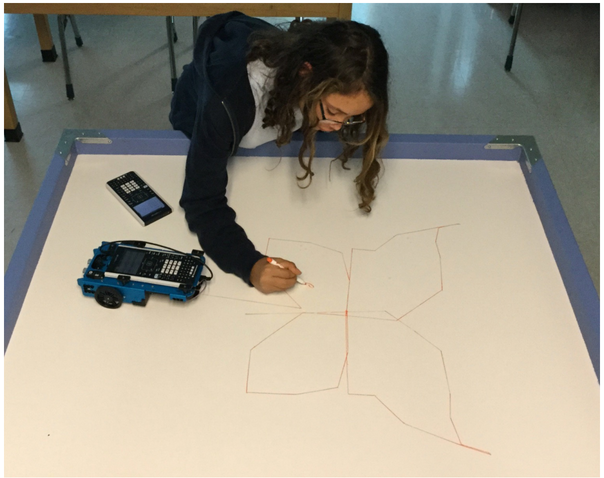
Where can you go next with TI-Rover and Dash?



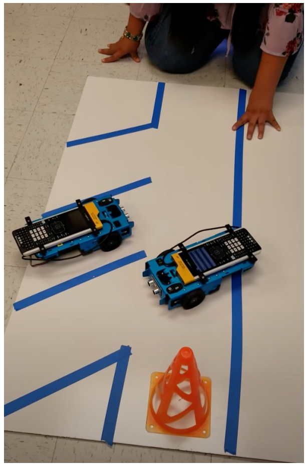
Drive an obstacle course



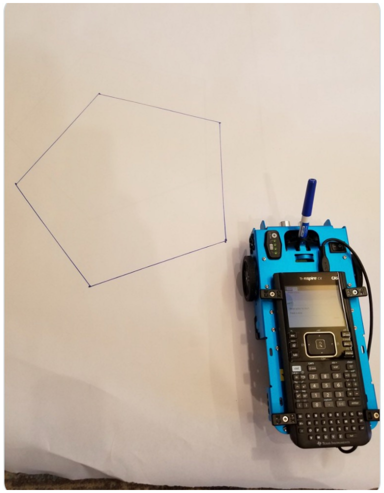
Drive a design



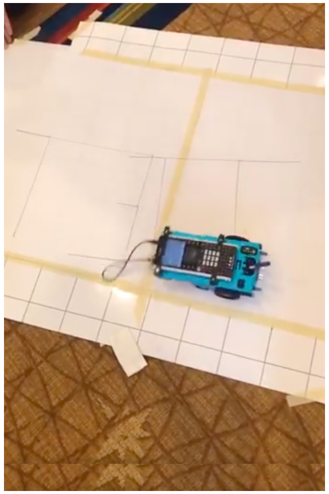
Draw artwork



Park your Rover



Use a For loop to draw polygons

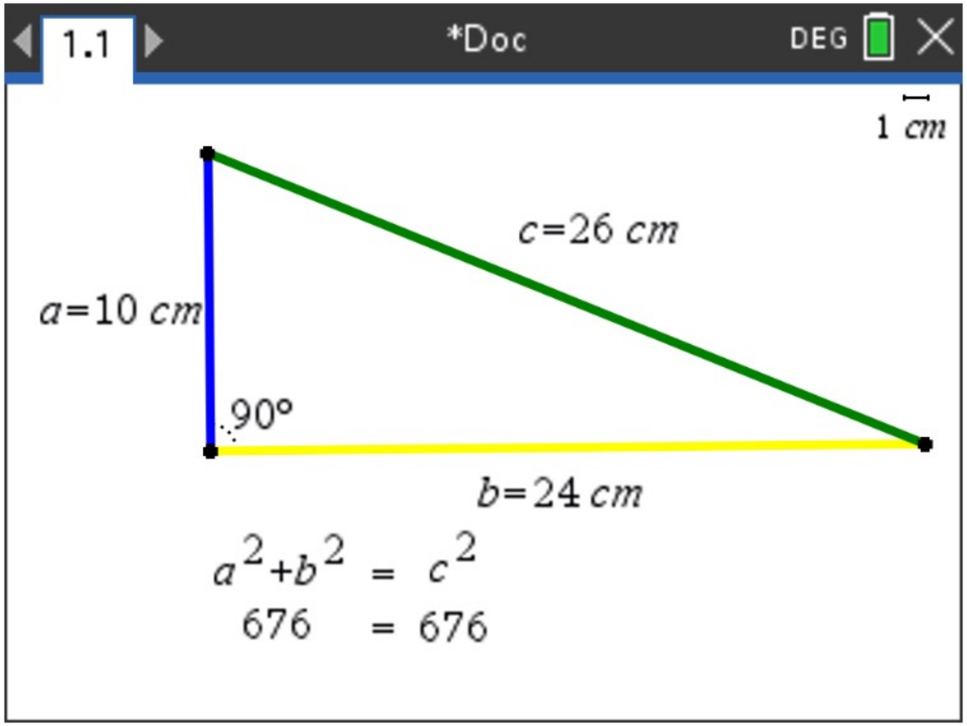


Write your name

Navigate a map

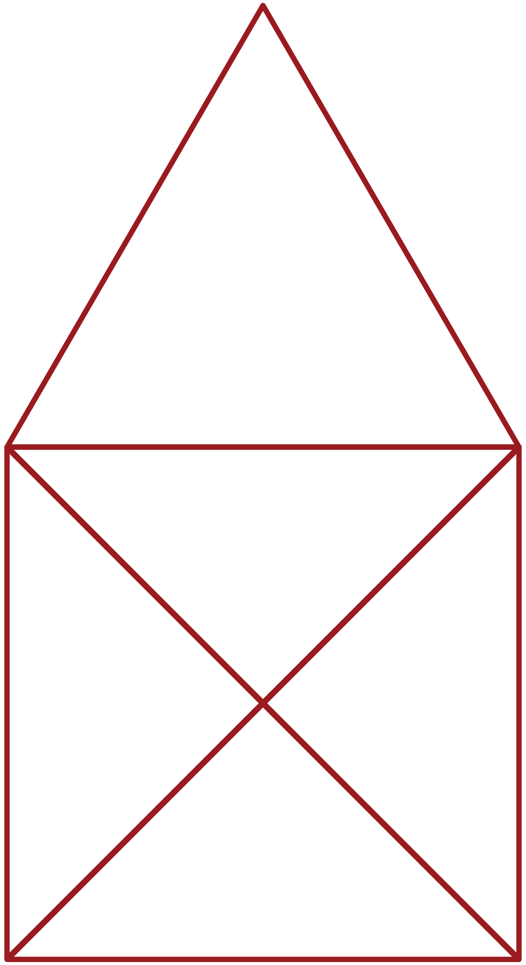
Quick Math Reminders

» Pythagorean Theorem



$$a^2+b^2 = c^2$$
$$676 = 676$$

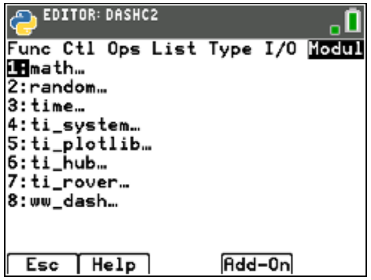
Logic Challenge 2

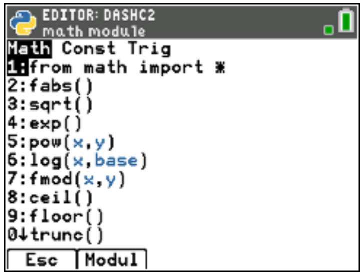


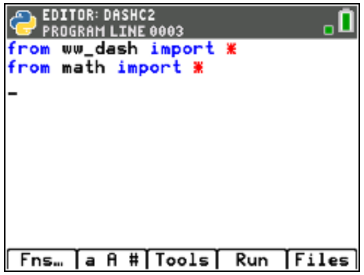
Task: Draw the figure shown large enough for Dash to drive.

Write a program to have your Dash drive the figure without crossing any lines.

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



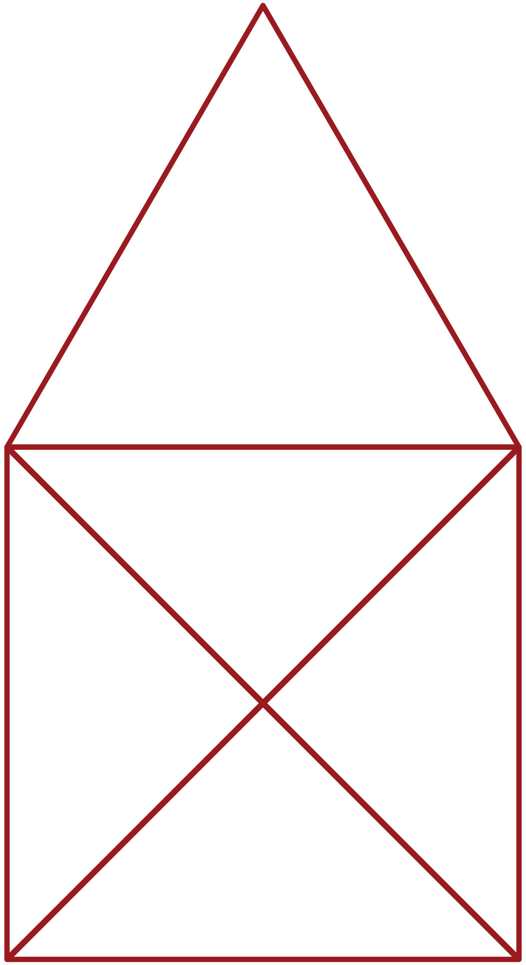




```
from ww_dash import *
from math import *
```

Marker version

Logic Challenge 2

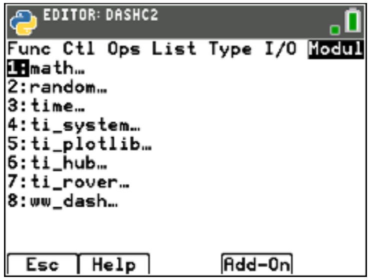


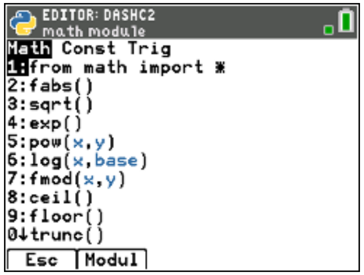
Task: Drive the figure shown without crossing any lines or going back over a line and without picking up the pen.

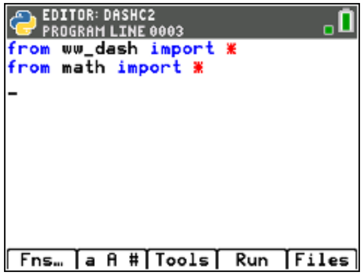
When you are ready, put the pen in and trace your path.

Note: For more information about marker holder 3D Print file see the earlier slide in this deck.

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

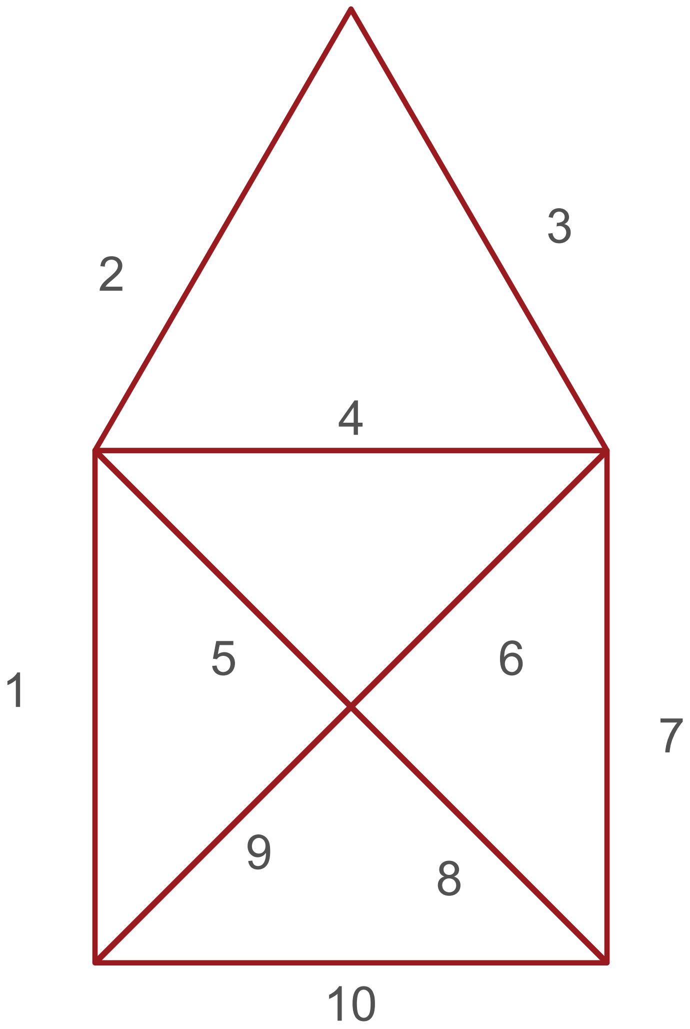




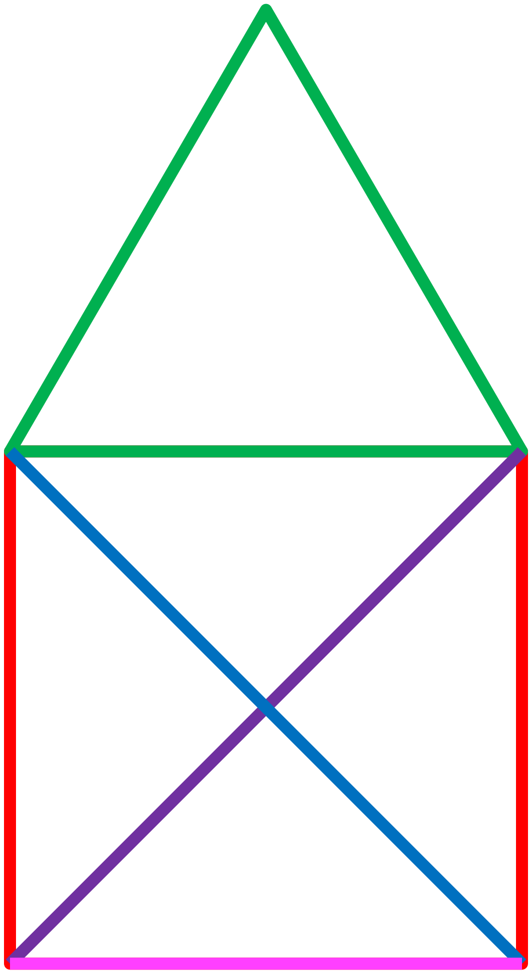


```
from ww_dash import *
from math import *
```

Logic Challenge 2 – example solution



Logic Challenge 3

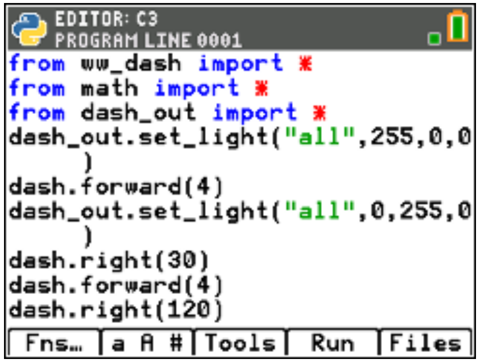


Task: Drive the figure shown without crossing any lines or going back over a line.

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

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

Put the set_light color statement before the drive statement that you want to match.
Note: The LED's stay the same color until another set_light color statement is run.



```
from ww_dash import *
from math import *
from dash_out import *
dash_out.set_light("all",255,0,0
      )
dash.forward(4)
dash_out.set_light("all",0,255,0
      )
dash.right(30)
dash.forward(4)
dash.right(120)
```

Thank You

www.TIstemProjects.com

Contact stem-team@ti.com with questions