

After successfully completing mini-project 1 your TI-84 Plus CE-to-Tello system is properly set up and ready to fly. In this project your Tello will depart its landing zone and take a tour of your room based on your measurements.

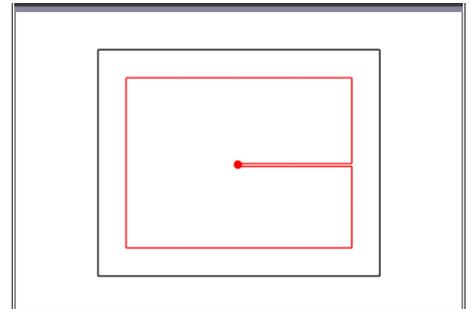
Remember the caution from project 1: the surface that Tello flies over needs to be a varied pattern. If the 'floor' is a smooth, uniform color then Tello will have difficulty maintaining its 'Vision Positioning System' and could possibly wander off course. And be sure the flight area (the entire room) is well-lit so that Tello can 'see' the ground. Keep a watchful eye on Tello as it flies around the room and be prepared to 'pick up the pieces'. Tello does not look forward.



Also be aware of the surroundings. You do not want to crash into something or hit someone.

0. The room you are working in is likely in the shape of a rectangle. Let's start the Tello in the center of the room, fly toward one wall, then fly along the four walls and return to the center of the room. See the red route pictured here.

In order to perform this task, you need to **measure the dimensions of the room in feet**. Use a regular carpenter's tape measure to measure the room. Write down your measurements to use in the program.



1. Begin a new Python program (ours is called **teltour.py**). Import the Tello module from **<Fns...> Modul <Add On> from tello import \***  
**from tello import \***

and perform a pre-flight check of the battery level before flying:

**b = tello.battery( )**

found on **[math] tello drone... Data**. Use a variable in front of the equals sign. If the battery is too low (10% or less) Tello will refuse to take off. If you think there is not enough charge to complete your mission, you can stop the program at any time by pressing **[on]** on the keypad. Tello will eventually land.





### TI-84 PLUS CE PYTHON

- Since you measured the room in feet and Tello only accepts measurements in centimeters, **define** a function to convert feet to centimeters. Get

```
def ( ) :
```

from the <Fns...> menu.

Also, get the **return** statement from the same menu and be sure it is indented to be the last statement of the function. We fill in the details next...

- The function name can be any identifier, such as **f**. We prefer to use a meaningful name, so we'll use the name **ftcm** and use the argument **ft** to represent the incoming value, a distance measured in feet.

The **block** of the function is the formula that converts feet to centimeters:

$$\text{cm} = \text{ft} * 12 * 2.54$$

and the **return** statement uses the variable **cm** to send its value back to the program. The completed function is:

```
def ft_to_cm( ft ) :  
  cm = ft * 12 * 2.54  
  return cm
```

You will use this function in the Tello Fly methods that require a distance. Now to code the main flight plan.

- For simplicity in running the program many times to work out the kinks in the flight, assign the dimensions of the room to two variables: **w** for width and **le\*** for length. The room we are using is 14 feet wide and 20 feet long. Your room will likely be different, but the algorithm should apply to any room size.

*Notes:*

*\* We use the identifier **le** because **l** (lowercase 'L') can be confused with the letter 'I' and the number 1. **ln** is the natural logarithm and **len** is a built-in length function.*

*To make a more user-friendly program you can use two input functions to enter these values, but who wants to type in two numbers every time you test the program?*

*Tello's maximum distance for a leg is 500cm, or about 16.4 feet. If your room is larger, then limit the dimensions so that Tello does not exceed this travel limit in any one leg of the journey.*

```
EDITOR: TELTOUR  
PROGRAM LINE 0007  
from tello import *  
b=tello.battery()  
  
def ():  
  *  
  *return  
-
```

```
EDITOR: TELTOUR  
PROGRAM LINE 0005  
from tello import *  
b=tello.battery()  
  
def ftcm(ft):  
  *cm=ft*12*.54  
  *return cm  
-
```

```
EDITOR: TELTOUR  
PROGRAM LINE 0011  
from tello import *  
b=tello.battery()  
  
def ftcm(ft):  
  *cm=ft*12*2.54  
  *return cm  
# use your dimensions here...  
w=14  
le=20  
-
```

# 10 Minutes of Code: Python Modules

## TI-84 PLUS CE PYTHON

5. Recall that our flight plan has Tello start in the center of the facing one wall (**w** in our code). The front of Tello has the tiny camera lens for taking pictures. The back holds the battery. The Tello function **tello.forward(cm)** causes Tello to fly forward some distance. How far is it to the wall?

answer:  $\text{length}/2$

but we don't want Tello to crash into the wall, so we stop short of the wall, say leaving 2 feet of space to the wall:

$$\text{le} / 2 - 2$$

*Note: The drawing to the right was made with Turtle graphics.*

6. In your program, issue the takeoff function, then use the **forward** function, both found on **[math] > tello drone... > Fly**

The forward(distance) argument is a value in *centimeters*, but our distance to fly,  $\text{le} / 2 - 2$ , is in *feet* so we must use our conversion function as the argument:

**tello.forward( fcm( le / 2 - 2 ) )**

7. For testing purposes, temporarily use the **tello.backward()** function to return Tello to its starting location to make sure that Tello behaves properly. Use the same argument as in the **tello.forward()** function.

Then use the **tello.land()** function.

Remember that all **tello.fly** methods are on

**[math] > tello drone... > Fly**

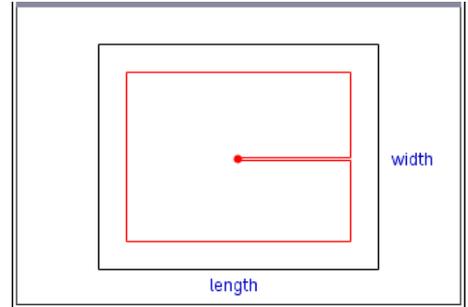
Double-check your code, stand back, make sure the flight path is clear, then **<Run>** the program.

8. If Tello returns to (approximately) its starting position, then you can complete the program to fly around the room. The next portion of the path travels along the wall (the bold red line seen here).

What is the length of this segment?

Answer:  $\text{width} / 2 - 2$

## TELLO TOURS THE ROOM



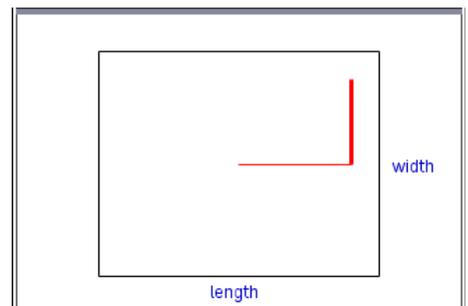
```
EDITOR: TELTOUR
PROGRAM LINE 0009
from tello import *
b=tello.battery()

def fcm(ft):
    cm=ft*12*2.54
    return cm

tello.takeoff()
tello.forward(fcm(le/2-2))
```

```
EDITOR: TELTOUR
PROGRAM LINE 0015
    cm=ft*12*2.54
    return cm
# use your dimensions here...
w=14
le=20

tello.takeoff()
tello.forward(fcm(le/2-2))
tello.backward(fcm(le/2-2))
tello.land()
```



### TI-84 PLUS CE PYTHON

9. At this point you, have two options: use `tello.turn_left(90)` and use `tello.forward( )` or just use `tello.fly_left( )`.

`tello.fly_left( distance )` makes Tello fly to its left *without* turning.

If it is safe to land in the corner, then use the `tello.land( )` function and test your flight now. And try both methods mentioned above as shown in the image as a *#comment*.

For the rest of the tour, you should stick with the method you use here: either (turning and forward) or just (flying in different directions).

10. The third leg of the trip is along the wall. How long is this segment?

Answer: **length – 4**

Remember that we must stay 2 feet from *all* walls.

11. Again, use either

`tello.turn_left(90)`

`tello.forward( distance )` assuming Tello is always flying forward or use

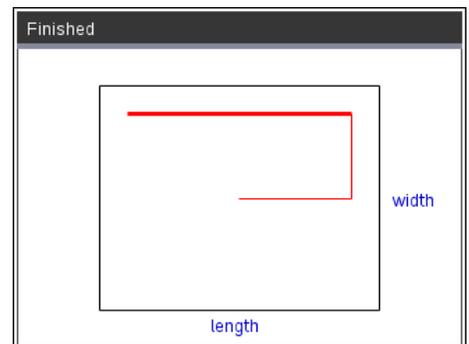
`tello.backward( distance )` assuming Tello is not turning.

In either case, the *distance* is `ftcm(le – 4)` determined in the previous step.

```

EDITOR: TELTOUR
PROGRAM LINE 0010
w=14
le=20

tello.takeoff()
tello.forward(ftcm(le/2-2))
#tello.backward(ftcm(le/2-2))
tello.turn_left(90)
tello.forward(ftcm(w/2-2))
# or use...
#tello.fly_left(ftcm(le/2-2))
    
```



```

EDITOR: TELTOUR
PROGRAM LINE 0020

tello.takeoff()
tello.forward(ftcm(le/2-2))
#tello.backward(ftcm(le/2-2))
tello.turn_left(90)
tello.forward(ftcm(w/2-2))
# or use...
#tello.fly_left(ftcm(le/2-2))
tello.turn_left(90)
tello.forward(ftcm(le-4))
    
```

TI-84 PLUS CE PYTHON

12. The complete flight plan using the `turn_` method is:

```

tello.takeoff()
tello.forward(ft_to_cm(le / 2 - 2))
#tello.backward(ft_to_cm(length / 2 - 2))
tello.turn_left( 90 )
tello.forward( ft_to_cm(w / 2 - 2) )
tello.turn_left( 90 )
tello.forward( ft_to_cm(le - 4) )
tello.turn_left( 90 )
tello.forward( ft_to_cm(w - 4) )
tello.turn_left( 90 )
tello.forward( ft_to_cm(le - 4) )
tello.turn_left( 90 )
tello.forward( ft_to_cm(w / 2 - 2) )
tello.turn_left( 90 )
tello.forward(ft_to_cm(le / 2 - 2))
tello.land()
    
```

If Tello's 'Vision Positioning System' works properly, Tello should have no trouble making this Grand Tour of your room!

13. To travel the same route *without turning* use the Tello flying methods `.forward( )`, `.fly_left( )`, `.backward( )`, and `.fly_right( )` in the proper sequence using the proper distances. The final program will be much shorter because there is no need to turn.



```

EDITOR: TELTOUR
PROGRAM LINE 0017
#tello.fly_left(ftcm(le/2-2))
tello.turn_left(90)
tello.forward(ftcm(le-4))
tello.turn_left(90)
tello.forward(ftcm(w-4))
tello.turn_left(90)
tello.forward(ftcm(le-4))
tello.turn_left(90)
tello.forward(ftcm(w/2-2))
tello.turn_left(90)
tello.forward(ftcm(le/2-2))
    
```

