

# TI-84 Plus CE Python Four Square



Tello Drone

## Activity Background and Learning Outcomes:

Students will master Python programming basics and fly the Tello drone using the [TI-84 Plus CE Python with the BBC micro:bit](#). Each challenge presents a new flight skill and promotes safety while flying the drone in the classroom. The objective of each challenge is to fly a square using different Tello methods and Python programming techniques.

Learning outcomes of this activity include:

### Python:

- editing a program
- running a program in shell
- types of variables
- defining and recalling variables
- basic input and output
- definite loops

### Tello:

- tello.takeoff()
- tello.land()
- tello.forward()
- tello.left()
- tello.right()
- tello.backward()
- tello.turn\_right()
- tello.turn\_left()
- tello.goto()



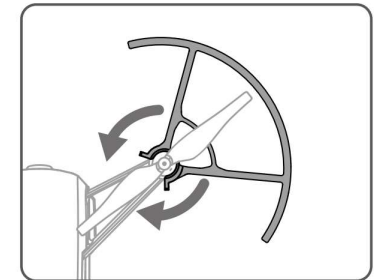
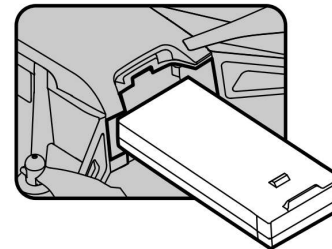
Scan QR code or click [here](#) to watch the associated videos for this activity

## Are you flight-rated for this activity?

Before proceeding with the activity, students should have completed the "Getting Your Wings" activity, and the Tello - micro:bit communication system is assembled and paired. If not, please complete the "Getting Your Wings" activity to prepare for this activity.

## Preflight Check: The Walk Around

1. Securely insert a fully charged battery into the drone. Be sure to slide the battery in with the correct orientation. **Never force the battery!**
2. Check that the propellers do not have nicks, and gently but firmly press each propeller onto the motor's shaft.
3. Confirm the propellers rotate freely and do not hit the safety guards.
4. Inspect the airframe for cracks or other damage.
5. Ensure the flight airspace is clear of people and objects.



## Reference Guide

### Python Syntax

takeoff()	tello.takeoff()
land()	tello.land()
forward( <i>distance</i> )	tello.forward(100)
backward( <i>distance</i> )	tello.backward(100 )
fly_right( <i>distance</i> )	tello.fly_right(100)
fly_left( <i>distance</i> )	tello.fly_left(100)
turn_right( <i>angle</i> )	tello.turn_right(90)
turn_left( <i>angle</i> )	tello.turn_left(90)
enable_mission_pad( <i>number</i> )	tello.enable_mission_pad(6)
 goto( <i>x,y,z</i> )	 tello.goto(50,75,100)
 print("string", <i>variable</i> )	 print("Side =",n)
 var = input("prompt")	 length = input("side length")
 var = float(var)	 length = float(length)
 for index in range(size): ◆◆ block	 for side in range(4): ◆◆tello.forward(100) ◆◆tello.right(90)

### Notes

Takeoff must be the first flight command in the program.

Land must be the last flight command in the program.

Fly forward at a distance from 10cm to 500cm.

Fly backward a distance from 10cm to 500cm.

Fly right, with no turn, from 10cm to 500cm.

Fly left, with no turn, from 10cm to 500cm.

Rotate right at an angle from 0° to 360°.

Rotate left at an angle from 0° to 360°.

**(EDU only)** Enable the visual positioning system to recognize the numbered mission pad under the drone at takeoff. The rocket printed on the pad points in the positive X-axis direction. This command must be issued after takeoff().

**(EDU only)** Fly to the x, y, and z coordinates over the enabled mission pad.

Coordinates must be within -500cm to 500cm. Note that x,y, and z coordinates can't be set simultaneously between -20cm and 20cm.

The print() function will display a string and/or variable in the calculator's Python shell.

The input() function requires the prompt displayed in the shell and returns a string to the assigned variable.

If the string returned from the input() function must be a number, for example, for a Tello flight method, then the string must be converted to a number type. The float() function returns the argument variable as a floating-point number type.

The for-control structure repeats a block of code the number of times set in the size of the range() function. Notice the code block is indented by two spaces ◆◆ in the TI-84 Plus CE Python editor. This indentation is essential to group all of the commands that are controlled with the for statement. If a command comes after the for statement and is not indented, it will not be repeated in the control structure.

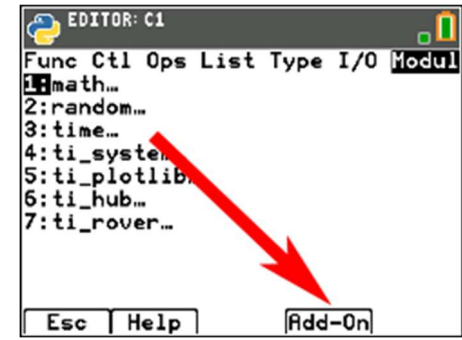
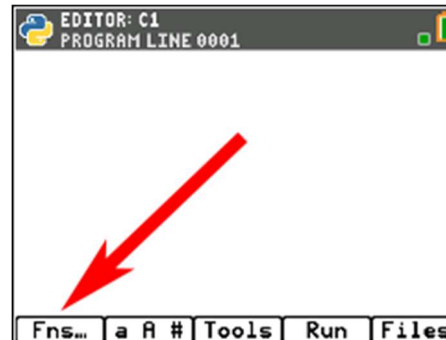
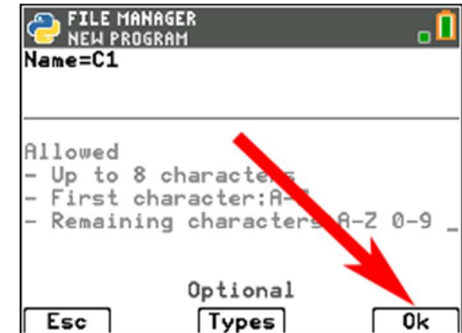
## Challenge 1: Square one

**The challenge:** Write a program to fly a square using only forward() and turn\_right() or turn\_left()

1. Select **[on]** key
2. Begin a new Python program by pressing the **[prgm]** key and selecting Python App.
3. Name the new program "C1", by pressing the **[C]** key and then **[2<sup>nd</sup>]**, **[alpha]** and **[1]**.
4. Select **Ok** to accept the name "C1".



5. Select the **Fns..** menu and then select **Modul.**
6. Select **Add-On**



7. Select **from tello import \*** to paste the statement into the editor.



*Note – to ensure your cursor goes to the following line after entering a parameter, press [2<sup>nd</sup>] + [enter]. This prevents the closed parenthesis from being dropped into the following line.*



Tello Drone

8. Select **Fns...** → **Modul** → **tello drone..** to access all of the Tello methods.

9. **Fns...** → **Modul** → **tello drone...** → **Fly** → **takeoff()**

*Note - The Tello methods will not appear in a menu until after the import statement is entered into the program.*

10. Select: **Fns...** → **Modul** → **tello drone...** → **forward(distance)**

11. Type in 100 and press [2<sup>nd</sup>] then [enter] to move cursor to following line.

12. Select: **Fns...** → **Modul** → **tello drone...** → **Fly** → **turn\_right(angle)**

13. Type in 90 and press [2<sup>nd</sup>] then [enter] to move cursor to following line.

```
EDITOR: C1
Fly Data Maneuver Set Commands
1:takeoff() check surroundings!
2:land()
3:forward(distance) 20-500cm
4:backward(distance) 20-500cm
5:turn_right(angle) 0-360°
6:turn_left(angle) 0-360°
7:up(distance) 20-500cm
8:down(distance) 20-500cm
9:fly_right(distance) 20-500cm
0↓fly_left(distance) 20-500cm
Esc Modul
```

```
EDITOR: C1
Fly Data Maneuver Set Commands
1:takeoff() check surroundings!
2:land()
3:forward(distance) 20-500cm
4:backward(distance) 20-500cm
5:turn_right(angle) 0-360°
6:turn_left(angle) 0-360°
7:up(distance) 20-500cm
8:down(distance) 20-500cm
9:fly_right(distance) 20-500cm
0↓fly_left(distance) 20-500cm
Esc Modul
```

```
EDITOR: C1
Fly Data Maneuver Set Commands
1:takeoff() check surroundings!
2:land()
3:forward(distance) 20-500cm
4:backward(distance) 20-500cm
5:turn_right(angle) 0-360°
6:turn_left(angle) 0-360°
7:up(distance) 20-500cm
8:down(distance) 20-500cm
9:fly_right(distance) 20-500cm
0↓fly_left(distance) 20-500cm
Esc Modul
```

```
EDITOR: C1
PROGRAM LINE 0003
from tello import *
tello.takeoff()
```

Fns... a A # Tools Run Files

```
EDITOR: C1
PROGRAM LINE 0004
from tello import *
tello.takeoff()
tello.forward(100)
```

Fns... a A # Tools Run Files

```
EDITOR: C1
PROGRAM LINE 0005
from tello import *
tello.takeoff()
tello.forward(100)
tello.turn_right(90)
```

Fns... a A # Tools Run Files

# TI-84 Plus CE Python Four Square



Tello Drone

14. Repeat steps 6 and 7 to complete flying a square returning to the original position.
15. When finished, land the drone **Fns...** → **Modul** → **tello drone...** → **Fly** → **land()**.

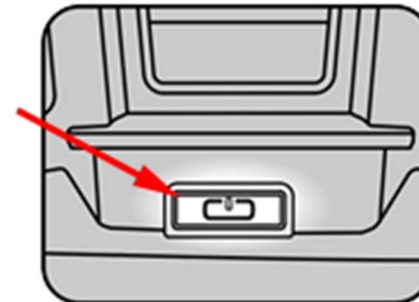
```
EDITOR: C1
PROGRAM LINE 0006
from tello import *
tello.takeoff()
tello.forward(100)
tello.turn_right(90)
tello.forward(100)
-
```

Fns... a A # Tools Run Files

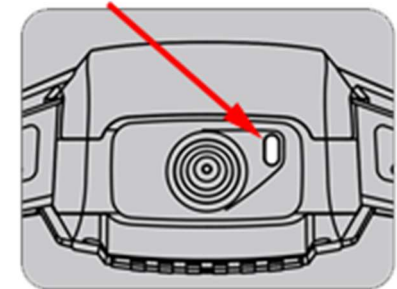
```
EDITOR: C1
PROGRAM LINE 0007
from tello import *
tello.takeoff()
tello.forward(100)
tello.turn_right(90)
tello.forward(100)
tello.turn_right(90)
```

Fns... a A # Tools Run Files

16. Set your drone on the floor on a Tello mission pad, clear from people and obstacles. Think about the program and where the drone will fly before launch.
17. Press the power button on the drone's side to turn it on.
18. Wait until the status LED on the front of the drone blinks steadily.



Power Button



Status LED

**Caution: Check that the flight path above and around the drone is clear of people and obstacles before running your program.**

19. Select **Run** to execute the program.
20. After the drone has landed, turn off the drone by pressing the power button again.
21. Congratulations! You have written the first fly-a-square program in Python!

```
EDITOR: C1
PROGRAM LINE 0011
from tello import *
tello.takeoff()
tello.forward(100)
tello.turn_right(90)
tello.forward(100)
tello.turn_right(90)
tello.forward(100)
tello.turn_right(90)
tello.forward(100)
tello.turn_right(90)
tello.land()_
```

Fns... a A # Tools Run Files

need image of shell after run



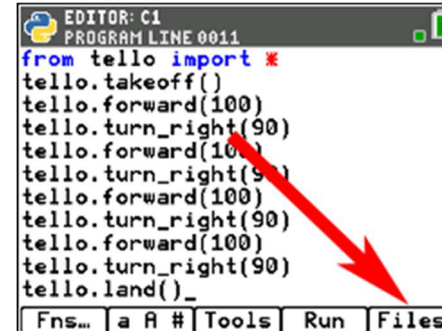


Tello Drone

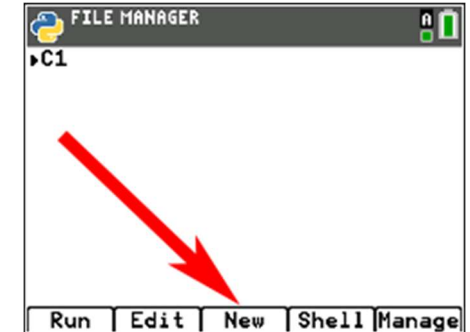
## Challenge 2: Square two

**The challenge:** Write a program name "c2" to fly a square with an edge length of 100 cm using only forward(), backward(), fly\_right(), and fly\_left() methods. Add a print statement after each side is flown with an appropriate message indicating which side of the square has been completed.

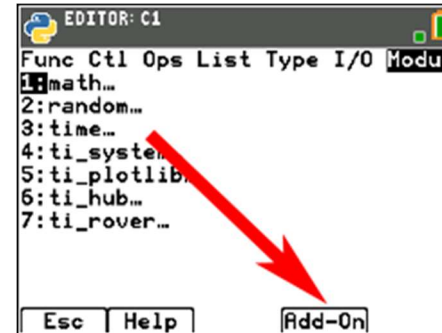
1. Start a new program. Select the **Files** menu and then **New**.
2. Name the new program "C2", by pressing the [C] key and then [2<sup>nd</sup>], [alpha] and [2].
3. Select **Ok** to accept the name "C2".



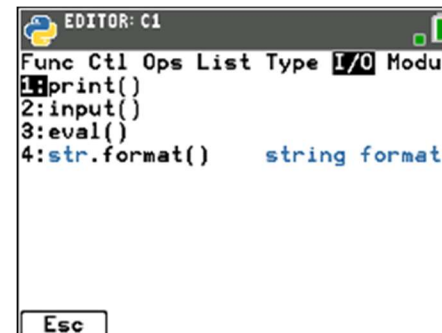
```
EDITOR: C1
PROGRAM LINE 0011
from tello import *
tello.takeoff()
tello.forward(100)
tello.turn_right(90)
tello.forward(100)
tello.turn_right(90)
tello.forward(100)
tello.turn_right(90)
tello.land()
```



4. Select: **Fns... → Modul → Add-On**
5. Select **from tello import \*** to paste the statement into the editor.
6. Select: **Fns... → Modul → tello drone... → Fly → takeoff()**
7. Select: **Fns... → Modul → tello drone... → forward(distance)**
8. Type in 100 and press [2<sup>nd</sup>] then [enter] to move cursor to following line.



9. Select: **Fns... → I/O → print()**



# TI-84 Plus CE Python Four Square



Tello Drone

10. Select the [2<sup>nd</sup>] and then [alpha] to lock the keypad into alpha mode; press the **quote** (addition) key to enter a single quote.
11. While the keypad is locked in the alpha mode, enter the letters for the message "side 1 has completed".



Note: The editor displays the text within quotes in green to let the programming know it is a string.

12. End the line with one more quote.
13. Select: **Fns... → Modul → tello drone...**
14. Complete the challenge by selecting the necessary flight methods and printing the corresponding side number. Don't forget to land!
15. Prepare the drone for takeoff as done in challenge 1.

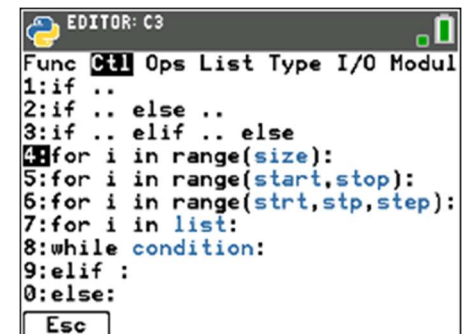
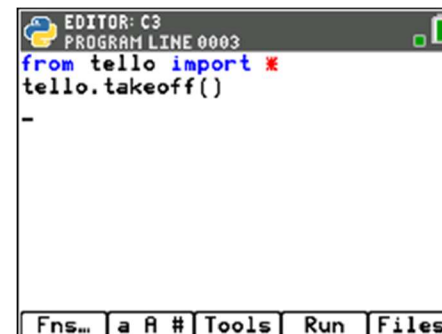
**Caution: Check that the flight path above and around the drone is clear of people and obstacles before running your program.**

16. Select **Run** to execute the program.
17. After landing, turn off the drone power by pressing the power button.

## Challenge 3: Square three

**The Challenge:** Use a *for index in range(size):* control loop with fly forward, turn, and a print statement displaying the side in the code block to fly a square with an edge length of 100 cm. When the square is completed, and the drone has landed, print a message indicating the drone has landed.

1. Start a new program. Select the **Files** menu and then **New**.
2. Name the new program "C3", by pressing the [C] key and then [2<sup>nd</sup>], [alpha] and [2].
3. Select **Ok** to accept the name "C3".
4. Select: **Fns... → Modul → Add-On**
5. Select **from tello import \*** to paste the statement into the editor.
6. Select: **Fns... → Modul → tello drone... → Fly→ takeoff()**
7. Select: **Fns... → Ctl → for i in range(size):**



# TI-84 Plus CE Python

## Four Square



Tello Drone

8. The index is a named variable that contains what step of the loop is running—in this example the default name “i” is used. If desired, the “i” can be edited and replaced with an alternate name.
9. The size parameter for the range() function is the number of times the loop will run. Since we are flying a square with four sides, the range is four.

10. Press the [↓] arrow down key to move the cursor to the next line.
11. The cursor is now indented; notice the two gray diamonds. Any code written in the indented block will be repeated in the loop. To fly a square, enter the code to fly forward 100 cm and then turn 90 degrees. Since these two methods will be repeated four times, the loop will fly a square!

Note: The *for index in range(size):* structure makes flying a square much easier than in Challenge 1 since the instructions for flying a side are repeated four times! Loops are a powerful coding tool.

12. Add a print statement to the block of code.
13. Type in "side", be sure to close the quote and then enter a comma outside of the second quote to add a second parameter to the print() statement. Notice the quote is green, and the comma is black. This color coding helps avoid syntax errors.

```
EDITOR: C3
PROGRAM LINE 0003
from tello import *
tello.takeoff()
for i in range(_:
**
```

```
EDITOR: C3
PROGRAM LINE 0003
from tello import *
tello.takeoff()
for i in range(4):
**
```

```
EDITOR: C3
PROGRAM LINE 0004
from tello import *
tello.takeoff()
for i in range(4):
**
-
```

```
EDITOR: C3
PROGRAM LINE 0006
from tello import *
tello.takeoff()
for i in range(4):
**tello.forward(100)
**tello.turn_right(90)
**
```

```
EDITOR: C3
PROGRAM LINE 0006
from tello import *
tello.takeoff()
for i in range(4):
**tello.forward(100)
**tello.turn_right(90)
**print(_
```

```
EDITOR: C3
PROGRAM LINE 0006
from tello import *
tello.takeoff()
for i in range(4):
**tello.forward(100)
**tello.turn_right(90)
**print("side",_
```



# TI-84 Plus CE Python Four Square



Tello Drone

14. The index variable named "i" keeps track of which step the loop is running. This variable can be printed to indicate the side of the square being flown. However, In Python, the first step of a loop is zero. Since numbering the first side of a square as "side 0" is awkward, a value of 1 should be added to the index variable. This expression, i+1, will add one to the index and display one when the index is zero.
  15. After the expression, type a comma to add a third parameter.
  16. Use quotes to add the string "completed" to the print statement.
  17. Notice the cursor remains indented with two diamonds, indicating this code line is within the loop. Since the drone should not land after the first step of the loop, the cursor needs to be dedented to the left to exit the loop before landing. Press [del] to delete the gray diamonds and dedent the cursor to the left to exit the loop.
  18. When all four steps of the loop have been completed. Don't forget to land the drone by using tello.land(). Be sure the method is not indented and part of the loop.
  19. Lastly, print a message reporting the drone has landed.
- Caution: Check that the flight path above and around the drone is clear of people and obstacles before running your program.**
20. Select **Run** to execute the program.
  21. After landing, turn off the drone power by pressing the power button.

```
EDITOR: C3
PROGRAM LINE 0007
from tello import *
tello.takeoff()
for i in range(4):
    tello.forward(100)
    tello.turn_right(90)
    print("side",i+1,"completed")
    -
```

```
EDITOR: C3
PROGRAM LINE 0007
from tello import *
tello.takeoff()
for i in range(4):
    tello.forward(100)
    tello.turn_right(90)
    print("side",i+1,"completed")
-
```



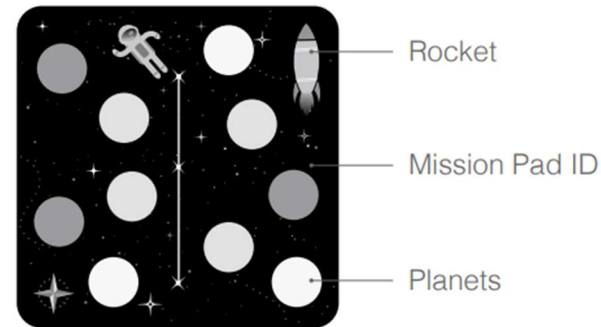
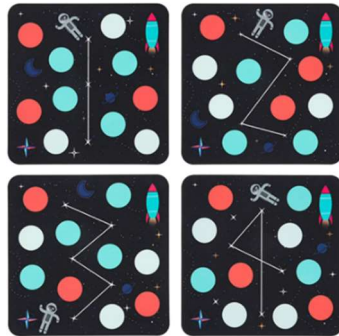
```
EDITOR: C3
PROGRAM LINE 0008
from tello import *
tello.takeoff()
for i in range(4):
    tello.forward(100)
    tello.turn_right(90)
    print("side",i+1,"completed")
print("Houston, the drone has la
nded")
```

Note: To enter a space in a string use [alpha] then [ ] as shown on right.

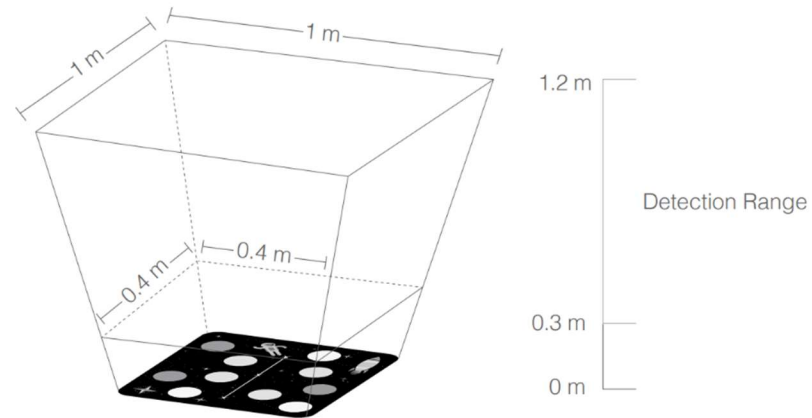
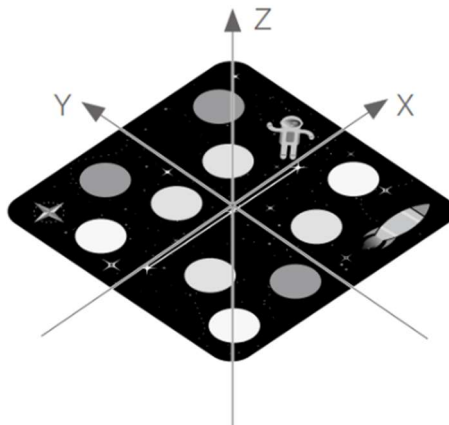
### Challenge 4: Square four

**The Challenge:** Use a mission pad to fly a square, with the size set by user input, within Tello's 3-D grid system. Use the `tello.goto(x,y,z)` method and `var = input("prompt")` function to complete the challenge.

**Background:** There are four Tello mission pads; each has an ID, rocket, and planets printed on one side of the pad that is recognized by the drone's vision system. The pads 1, 2, 3, and 4 are shown below. The pads establish a 3-dimensional coordinate system the drone can fly within.



The rocket in the corner of the pad points in the positive X-axis direction and defines the 3-D coordinate system for the drone. The center of the pad is the origin (0,0,0) of the system. The pad has a zero height on the Z-axis; however, the drone will not fly lower than  $z = 30\text{cm}$  to prevent crashes into the floor. Limiting the `goto(x,y,z)` values within this recommended volume, see diagram below, is best. The mission pads only work with the Tello EDU and TT models.



# TI-84 Plus CE Python Four Square



Tello Drone

1. Start a new program. Select the **Files** menu and then **New**.
2. Name the new program "C4", by pressing the **[C]** key and then **[2<sup>nd</sup>]**, **[alpha]** and **[2]**.
3. Select **Ok** to accept the name "C4".
4. Select **Fns... → Modul → Add-On**
5. Select **from tello import \*** to paste the statement into the editor.
6. The input function prompts the user to enter something, in this program, the length of the square's edge in cm. The function returns the user's keyboard entry as a string. The returned string must be stored in a variable for later use. In the example code, the variable is named "side". Use the keypad to type in the variable name, "side", and then type in the equals, "=", symbol.
7. Select: **Fns... → I/O → input()**
8. The input function requires a prompt string as an argument; this string will be displayed on the calculator and will tell the user to enter. Within the parentheses, select the **[ctrl]** and then the **"□"** key to enter a single quote. Use the keypad to type in "enter side length in cm" with a space after the "m" be sure to close the quotation.
9. The input() function returns a string into the variable named "side"; forcing the "side" variable to be a string type. String type variables may contain any keyboard character, including numbers. However, string types differ from number types. It must be a number type to do any mathematical operations on a variable. There are two number types; integers such as 1,2, or 3, and floating point such as 4.5, .0023, or 10.0. The Tello fly method arguments require a floating-point type, and a string type must be converted to a float type. The float () function can redefine the "side " variable. The name "side" should not be in quotes since it is the name of the variable not defining a string.
10. Use the keypad to type "side =".
11. Select **Fns... → Type → float()**

```
EDITOR: C4
PROGRAM LINE 0002
from tello import *
-
```

Fns... a A # Tools Run Files

```
EDITOR: C4
PROGRAM LINE 0002
from tello import *
side=
```

Fns... a A # Tools Run Files

```
EDITOR: C4
Func Ctl Ops List Type I/O Modul
1:print()
2:input()
3:eval()
4:str.format() string format
```

Esc

```
EDITOR: C4
PROGRAM LINE 0003
from tello import *
side=input("enter side length in
cm")
-
```

Fns... a A # Tools Run Files

```
EDITOR: C4
PROGRAM LINE 0003
from tello import *
side=input("enter side length in
cm")
side=
```

Fns... a A # Tools Run Files

```
EDITOR: C4
Func Ctl Ops List Type I/O Modul
1:int()
2:float()
3:round()
4:str()
5:complex(real,imag)
6:type()
```

Esc



Tello Drone

12. Use the keypad to type in the name of the variable "side" within the parenthese. This line of code redefines the "side" variable from a string type to a floating-point type.

Note: The name of the variable should not be in quotes since it is the name of the variable and is not defining a string.

13. Select: **Fns...** → **Modul** → **tello drone...** → **Fly** → **takeoff()**

14. Select: **Fns...** → **Modul** → **tello drone...** → **Fly** → **enable\_mission\_pad(number)**

15. Choose a Tello mission pad to use for your flight. In the example code, mission pad 3 was used. Be sure to enter the id of the mission pad you will use on your flight.

Note: The `tello.enable_mission_pad()` method must be entered after the `tello.takeoff()` method; this allows the drone to "see" the pad from above.

16. Imagine the drone flying the square on the left with a side = 100 cm.

Make a list of the coordinates of each vertex of the square.

X1, Y1 = ( 75 , 0 )

X2, Y2 = ( , )

X3, Y3 = ( , )

X4, Y4 = ( , )

Make a similar list of the coordinates of each vertex of the square using the variable side.

X1, Y1 = (side , 0 )

X2, Y2 = ( , )

X3, Y3 = ( , )

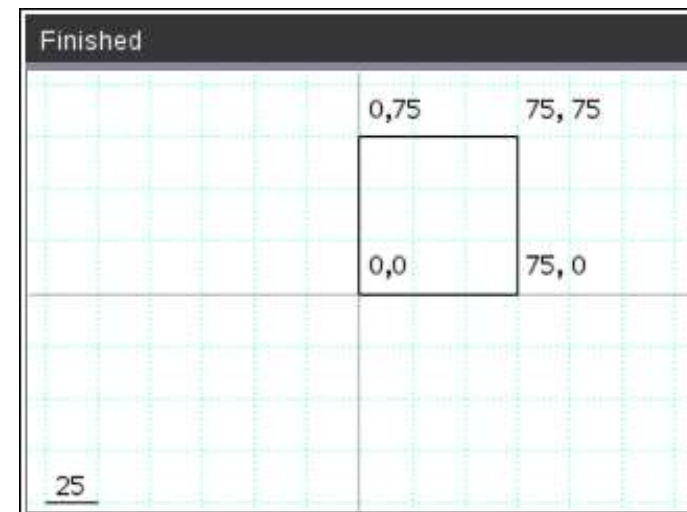
X4, Y4 = ( , )

```
EDITOR: C4
PROGRAM LINE 0004
from tello import *
side=input("enter side length in
cm")
side=float(side)
-
```

```
EDITOR: C4
PROGRAM LINE 0005
from tello import *
side=input("enter side length in
cm")
side=float(side)
tello.takeoff()
-
```

```
EDITOR: C4
Fly Data Maneuver Set Commands
5:turn_right(angle) 0-360°
6:turn_left(angle) 0-360°
7:up(distance) 20-500cm
8:down(distance) 20-500cm
9:fly_right(distance) 20-500cm
0:fly_left(distance) 20-500cm
A:forward_time(time) 1-10s
B:backward_time(time) 1-10s
enable_mission_pad(number) ID
D:goto(x,y,z) cm
```

```
EDITOR: C4
PROGRAM LINE 0006
from tello import *
side=input("enter side length in
cm")
side=float(side)
tello.takeoff()
tello.enable_mission_pad(3)
-
```



TI 84 Plus CE -Python turtle program showing the flight path

# TI-84 Plus CE Python Four Square



Tello Drone

17. Select: **Fns...** → **Modul** → **tello drone...** → **Fly** → **goto(x,y,z)**
18. Use the number pad to type in the x,y, and z coordinates for the first vertex of the square. The example code uses a height of 50 cm, z=50.

Note: The drone can't fly below 30 cm. All (x,y,z) triplets must have a z value greater than 30.

19. Repeat steps 16-18 for the other three vertices of the square.
20. When completed, land the drone.
21. Center your drone on the mission pad and test your program!

**Caution: Check that the flight path above and around the drone is clear of people and obstacles before running your program.**

22. Select **Run**. After landing, turn off the drone power by pressing the power button.

Fly	Data	Maneuver	Set	Commands
5	turn_right	(angle)	0-360°	
6	turn_left	(angle)	0-360°	
7	up	(distance)	20-500cm	
8	down	(distance)	20-500cm	
9	fly_right	(distance)	20-500cm	
0	fly_left	(distance)	20-500cm	
A	forward_time	(time)	1-10s	
B	backward_time	(time)	1-10s	
C	enable_mission_pad	(number)	ID	
D	goto	(x,y,z)	cm	

Esc Modul

```

EDITOR: C4
PROGRAM LINE 0007
from tello import *
side=input("enter side length in
cm")
side=float(side)
tello.takeoff()
tello.enable_mission_pad(3)
tello.goto(side,0,50)
-
  
```

Fns... a A # Tools Run Files