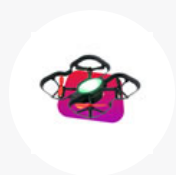


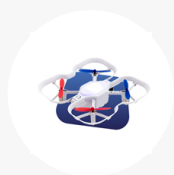
## 1.1: First Flight

So you know how to fly the drone with a remote, but can you code it to fly as well? Check out the Goals and Steps to Success to see what you will learn in this lesson!

This lesson works with:



CoDrone EDU



CoDrone EDU  
(JROTC ed.)



Python

beginner

CDE

CoDrone EDU

flight events

fundamentals

landing

Python

takeoff

Grade level:

6 - 12+

Approx. time required:

30 - 45 mins

### Step 1

## Goals and Steps to Success

By the end of this lesson, you should be able to:

- Use the print function and describe its value in debugging (finding and fixing errors) in your code
- Add comments to your code and understand the importance of well documented code
- Pair with your drone using the pair() function
- Understand how to use the takeoff() and land() functions

In this lesson, you will complete the following steps. Make sure that you demonstrate your completed code to your instructor after each step:

1. Hello World! (Using the print() function)
2. Using Comments

3. Pairing and Printing
4. Takeoff! (Your First Flight Test)

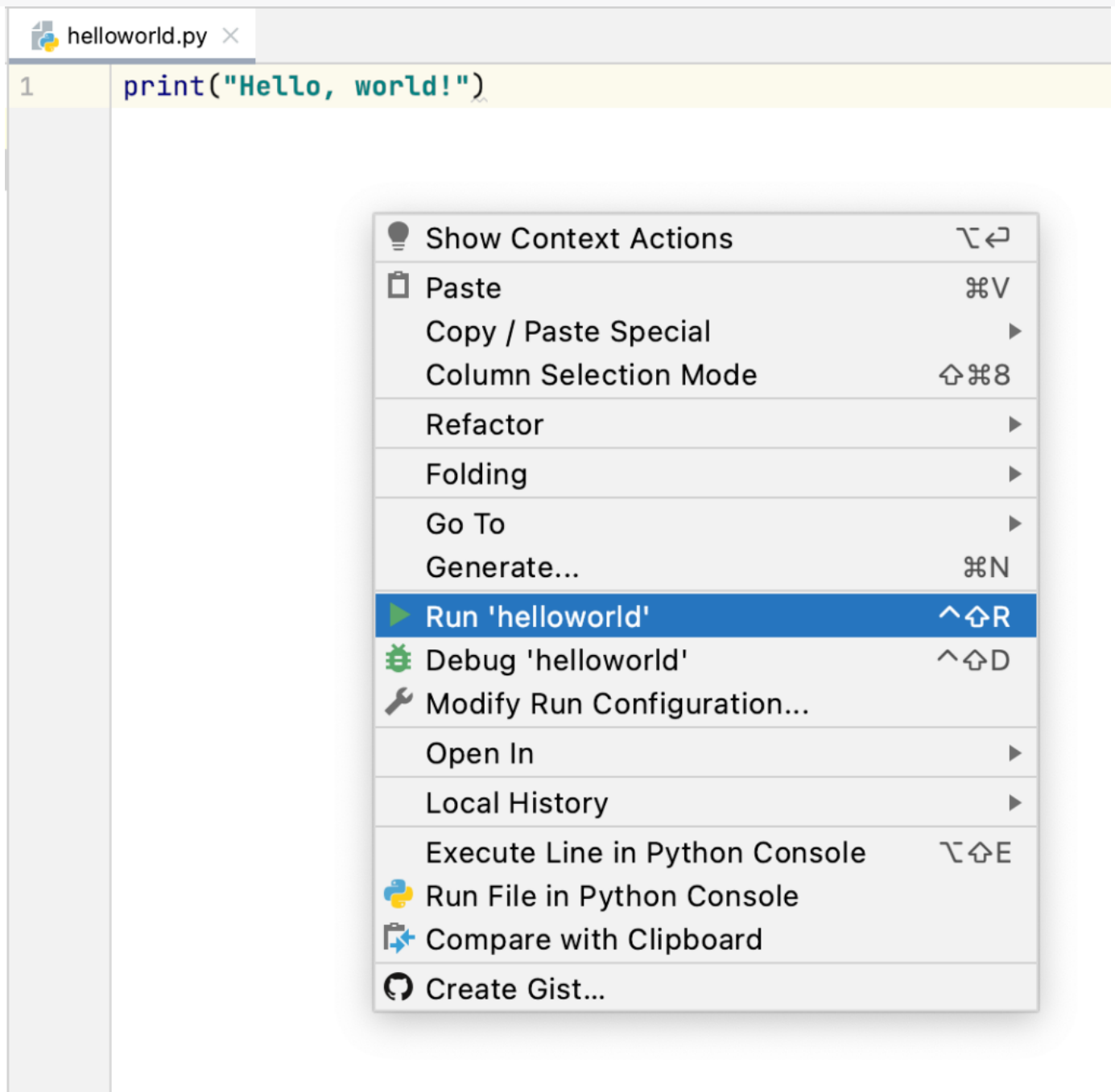
## Step 2

# Hello, World!

It's time to write your first Python program! This program will use the `print()` function. You're not printing with ink and paper, but you'll print to the screen so you can see any output from the program. Programmers use print statements to determine where the program is when they run it (since you can't tell once you run the code) which helps them **debug** (find errors in) the code. They also use the `print()` function to check the values of variables (containers that store values) as the program progresses.

Make a new Python file called *helloworld.py* and run this line of code:

```
print("Hello, world!")
```



Print statements are sent to the Run console. The console is a tool in most IDEs (Integrated Development Environments) that allows you to see specific commands and information about the program as it runs. In pycharm, the console is just called Run and appears at the bottom of your screen.

Click “Run”



The “Hello World” program has been used to introduce beginner programmers to a new programming language since the 1970s. Welcome to Python!

## Step 3

### Using Comments

Comments are ignored by the computer when running your code, so you can write words and sentences as you would in a normal document. You can add comments to your code by beginning the line with a hash symbol (`#`). These are for single-line only comments, which means that when you start a new line, the program goes back to normal code.

Try it out with the code below!

```
# print("The computer is ignoring this")
```

```
print("The computer is printing this!")
```

What do you see?

If you want to write a longer comment, insert the block of code between two sets of triple quotation marks or triple apostrophes. The first set of triple apostrophes is the beginning of the comment and the second set is the end of the comment. Everything in between will be ignored by the computer. Try it out with the code below!

```
'''
```

```
This is a longer comment that the computer will not read:
```

```
I
Love
Programming
Drones!
'''
```

Comments can also be used by programmers to debug by “turning off” parts of the code by commenting them so they are ignored. This way they can test specific parts of a program to narrow down what the problem may be.

## Step 4

### Pairing and Printing

Now that we know how to use print statements, we will connect to the CoDrone EDU and debug our program using print statements. Before we get started, please complete the following steps:

1. Make sure you have imported the CoDrone EDU library in project interpreter.
2. Click on the “File” menu and choose “New”
3. Choose Python File
4. Give the new file the name *takeoff.py*
5. Click “Ok” or press Enter
6. A new tab in Pycharm should be created that has our program name *takeoff.py* as the tab name

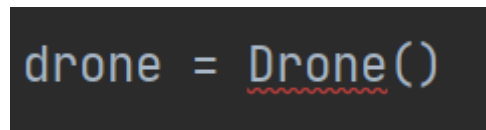
Now that you have those steps completed, you will always need to tell python you want to use the CoDrone EDU library. Think of the library like a recipe book that your program needs to be able to code your drone! Inside the library there are all sorts of functions, or “recipes”, that you will use. You will learn more about functions later. To do this you will use the **import** statement. This tells python that you want to use the tools in a specific library. In this case, you want to use the tools from the CoDrone EDU’s library so you can program it.

Import the CoDrone EDU library into your program with the following line of code:

```
from codrone_edu.drone import *
```

**IMPORTANT NOTE:** In future programs, if you forget to include the import

statement, Pycharm will give you an error which looks like a red squiggly line whenever you try to create the drone in the program (as shown in the image below), so make sure you always remember to include this line!



```
drone = Drone()
```

After you import the library, you need to create an object, which is a collection of variables and functions that act on data. Create an object called “drone” like this:

```
drone = Drone()
```

In this example, CoDrone EDU is a class. You do not have to understand the details now, but just know that a class is like a factory that can make CoDrone EDU objects. All the objects have the same characteristics and the same accessible functions.

Now you need to pair your drone with this command line:

```
drone.pair()
```

The pair function will find your controller automatically.

Finally, add one more line that will close the connection. This is an important step because if the communication channel with the drone is not closed, the next time you try to upload code, it will still be looking for the old connection. So make sure you close the connection at the end of each program.

```
drone.close()
```

So far your program should look like this:

```
from codrone_edu.drone import *  
drone = Drone()  
drone.pair()  
drone.close()
```

Now, try to add print statements to let you know the following:

1. When the drone object has been created
2. When the drone is ready to pair
3. When the drone has successfully paired.

As we discussed before, using print statements can help us make sure that all parts of our program successfully execute. If you are having trouble, see the

example code below:

```
from codrone_edu.drone import *
print("Creating drone object")
drone = Drone()
print("Getting ready to pair")
drone.pair()
print("Paired!")
drone.close()
```

## Step 5

### Takeoff! (Your First Flight Test!)

In this step, we will learn how to make the drone take off and land. Since the drone will be flying, please make sure you follow all safety precautions and make sure you place your CoDrone EDU away from other objects on the floor.

The `takeoff()` command



Follow the steps below to prepare a new file for this step:

1. Create a file called *flight\_test.py* (note we use an underscore, you could also use a dash-)
2. In this new file, add a drone object (you can call it `drone` or give it a name of your choosing)
3. Tell the drone object to pair

We use an underscore or dash to make the file name readable. It is possible to use a space, but is not good programming practice as not all interpreters will be able to understand or use file names with spaces. If you are having trouble with these steps, please go back and review the previous step and code.

Now we are ready to write our flight test! Write the `drone.takeoff()` command, which will allow your CoDrone EDU will take off and hover. Note that the takeoff command does not take in any **parameters**, so the parenthesis should be empty. This is because `drone.takeoff()` goes through the steps of starting the motors and flying the drone to a preset height. Add the following code to your *flight\_test.py*:

```
drone.takeoff()  
print("taking off")
```

The `land()` function



Much like the `takeoff()` function, the `land()` function follows a preset routine that will land the drone safely on the ground.

There are two ways a drone can land. The first is with the `drone.land()` command, which will gently bring your CoDrone EDU to the ground.

```
drone.land()
```

The second is with the `drone.emergency_stop()` command. This immediately turns off all the motors, which will crash your CoDrone EDU to the ground. Ideally, this should only be used in emergency situations, and you can program your keyboard's spacebar to act as an emergency stop if it's hit. For now, we will stick with a simple land command. Below you will find an example of the code with print statements included.

```
from codrone_edu.drone import *
```

```
drone = Drone()  
drone.pair()  
print("Paired!")  
drone.takeoff()  
print("In the air!")  
print("Landing")  
drone.land()
```



```
drone.close()  
print("Program complete")
```

## Step 6

# Lesson Complete

In this lesson, you learned about how to takeoff and land your drone just by using code! In the next lesson, learn how to use the `hover()` function to fly in place.

