Turtle Graphics with Python!

A Teknowledge Activity

# Basics

Go to [bit.ly/tkturtle](http://bit.ly/tkturtle) in your internet browser.

For now, skip the first two lines of code (You can look at the last page in this handout, under “Fun Facts #1”, if you’re curious about what they do).

Below that, we have:

|  |  |
| --- | --- |
| 4  5  6 | turtle.Screen().bgcolor("black")  turtle.color("white")  turtle.shape("turtle") |

line #4 - sets the screen background color to black

line #5 - sets the turtle color to white

line #6 - sets the turtle to be shaped like a turtle

If you put different words inside the parenthesis and quotation marks, you can change what these lines of code will do!

**Challenge 1: Change the background color and turtle color to colors that you like!**

Tips:

* Python can recognize lots of colors other than black and white! Lines #4 and #5 can be changed to make the background or turtle a different color.
* You can also choose a different shape! Try replacing the word “turtle” on line 6 with one of these: “arrow”, “turtle”, “circle”, “square”, “triangle”, “classic”.
* Make sure that your capitalization is right when you are typing things! Sometimes programs won’t recognize things that you type if some letters are capitalized that they don’t expect.

Keywords:

* A **function** is a command that you can use to tell the program what to do. Each time you gave your partner directions, like “walk forward”, you were kind of using a function. Usually, each function goes on its own line.
* To **call** a function is just another way to say “use” a function. In each line above, you “called” a different function. This can also be used like a noun. Each line was a “call” to a function.
* An **argument** is something that you write inside the parenthesis when you are calling a function. The argument for the first function was “black”, and now it is whatever color you chose instead. When you were giving directions to your partner, the *argument* would be *how many* steps to walk or *how much* to turn.
* To **pass** an argument means to use it when you’re calling a function.

Using these keywords all together, we can say:

You **called** the “turtle.color” **function** on line 5, and **passed** the word “white” as its **argument**.

You **called** the “walk forward” **function** when talking to your partner, and **passed** “4 steps” as its **argument**.

# Let’s Get Moving

We can tell the turtle to go forward or backwards a certain number of pixels like so:

|  |
| --- |
| turtle.forward(50)  turtle.backward(100) |

The number passed into the functions “forward” and “backward” is the number of pixels to move.

**Challenge 2: Try moving the turtle off the screen forwards, then off the screen the other way, backwards, then back to the original spot.**

We can also tell the turtle to turn a certain number of degrees.

|  |
| --- |
| turtle.left(45)  turtle.right(360) |

**Challenge 3: Draw a square and a “+” using the turtle.**

Tips:

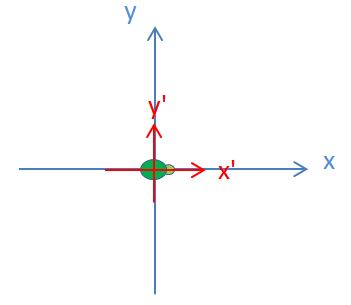
* If you’re stuck on what directions to give the turtle, try drawing the letter on a piece of paper without lifting it up. As you draw it, think about how far you are dragging the pen or pencil, and then what direction you have to turn in at each step.
* Remember that 360 degrees is a full circle. That means if you want to turn the turtle around in the opposite direction, you should turn 180 degrees. You can experiment with smaller degrees like 90 and 45.

# Now We’re Going Places: Coordinates

We can tell the turtle to go to a specific location like this:

|  |
| --- |
| turtle.goto(100, 50) |

The two numbers are (x, y) values in a coordinate plane like this.



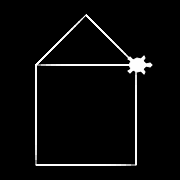
Positive numbers for x go **right**.

Positive numbers for y go **up**.

**(0, 0)** is in the middle of the screen.

**Challenge 4: Make the turtle draw a basic house (like the one shown below), using only turtle.goto commands, starting with this line of code:**

|  |
| --- |
| turtle.goto(50, 50) |



Tips:

* If you’re trying to decide what *coordinates* you should go to, try drawing it one step at a time. Experiment with different numbers and see if the turtle went where you expected.
* It doesn’t matter if the turtle has to go backwards over a line that’s already drawn, or if it ends up at a different spot than in the picture above.

# Pen and Stamp and Text Party

Sometimes you don’t want the turtle to draw a line when it moves.

You can “lift” and “lower” the “pen” with this code:

|  |
| --- |
| turtle.penup()  turtle.forward(20)  turtle.pendown()  turtle.forward(20) |

You can also “stamp” the turtle with this code:

|  |
| --- |
| turtle.stamp()  turtle.forward(100) |

And you can write text onto the screen with this code:

|  |
| --- |
| turtle.write("YEAH") |

If you want to get fancy, you can find advanced text writing options on the last page under “Fun Facts #2”.

**Challenge 5: With these new skills, try to create this image as close as you can with your turtle:**

****

Tips:

* Remember to use “Turtle.penup()” and “Turtle.pendown()” when you want to move the turtle without drawing on the way there.
* If you want the turtle stamps to be in the right direction, try some turning functions like “right(45)”.

You might be wondering how we could make this drawing happier with a smiling mouth instead, but drawing c*urved* lines using only these functions is more difficult. If you want an extra challenge, go to the last page under “Bonus Activities #1”, where you will try to draw a circle.

**Fun Facts**

1. Note the first two lines of code:

|  |  |
| --- | --- |
| 1  2 | #!/bin/python3  import turtle |

* line #1 - tells trinket.io we are using python3 (not python2).
* line #2 - gives us access to magic turtle powers in this coding file.
* The “import” word means that we want to enable a new “module” of commands for use in our code below. In this case, we are using the “turtle” module.

1. Turtle advanced text writing will look something like this:

|  |
| --- |
| turtle.write("YEAH", align="center", font=("Helvetica", 18, "bold")) |

* Try pasting this line into your code, and then changing the word “YEAH” or the number 18.
* This is an example of a function with *multiple* arguments. Notice that each of them is separated by a comma so that Python knows when you are done with one piece of information and you are starting on another one.

**Bonus Activities**

1. Using turtle, with just the functions “forward” and “right”, try to draw a circle.

* It will be difficult to make smooth edges but see how smooth you can get them to be.
* You might have to write a few lines of code to draw part of the circle, and then copy and paste lots of times to make the whole thing.
* In another lab, we will learn a way to do this with very few lines of code using something called a **loop**.