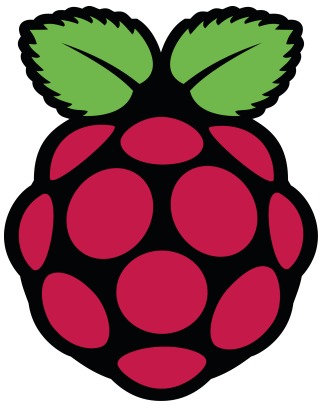
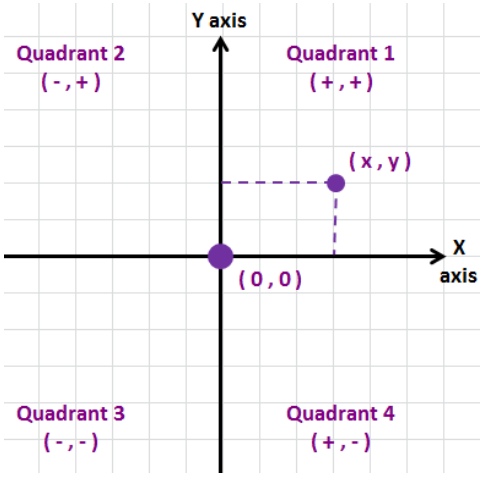


Extra Material: Turtle Basics

Using the Turtle to Draw Graphics on Your Raspberry Pi!

What's a Turtle?



Let's imagine that you have a tiny little robot, called a **turtle**, that lives on the X-Y coordinate plane. He begins at the point (0,0), or the **origin**, and can move forward, backward, in circles, or any other way you tell it to. With it, the turtle always carries a **pen**, and draws a little line behind it wherever it moves.

We can use the turtle to help us create some drawings on our Raspberry Pi! In this section, we'll cover the basics of how the turtle works.

Setting Up Your Turtle

Setting up your turtle is easy. To get started, all you have to do is write this at the top of your program:

```
import turtle
```

That's it! From there, you can use any one of Python's `turtle` functions, a collection of which I have listed on the next few pages.

First, let's try a quick example. Below is a short turtle program that I put together:

```
import turtle
turtle.forward(50)
turtle.color("red")
turtle.left(25)
turtle.forward(50)
```

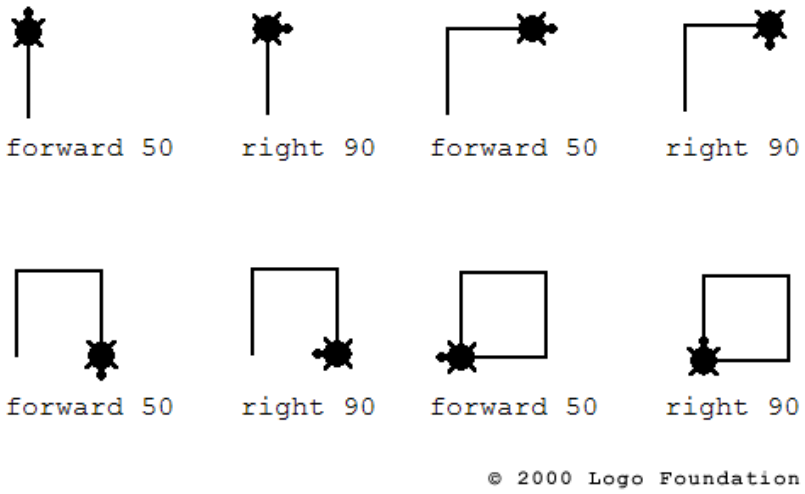
Try running this, and see how the turtle moves. The turtle always begins in the center of the page, *facing to the right*. From there, the turtle begins by moving forward 50. Then, it changes the color of the pen that it is holding to "**red**"! The turtle turns to the left by 25 degrees, and finally moves forward again by 50. Easy as that! Try changing around some numbers and study what the turtle can do!

Some Basic Turtle Commands

Below, I've listed a few basic turtle commands along with short descriptions on how to use them. Try playing around with these commands, and draw some pictures of your own!

1. Move your turtle forward or backward. *In the parentheses* **()**, *type in a number to represent the DISTANCE you would like to move.* An example:

```
turtle.forward(40)
turtle.backward(30)
```



2. Turn your turtle to the left or right. *In the parentheses* `()`, *type in a number to represent the ANGLE (in degrees) that you would like your turtle to turn. Tip: a RIGHT ANGLE is 90 degrees!* An example:

```
turtle.left(25)
```

```
turtle.right(50)
```

3. Tell your turtle to go back to its starting position.

```
turtle.home()
```

4. Change the turtle's drawing speed. *The number in the parentheses* `()` *must be an integer between 1 and 10. 1 is the slowest, while 10 is the fastest.*

```
turtle.speed(2)
```

5. Change the color of the line that the turtle draws. *In the parentheses* `()`, *type in the name of the color in quotes* `" "`.

```
turtle.color("blue")
```

Some Advanced Turtle Commands

Below is a long (but not exhaustive!) list of some more advanced turtle commands!

1. Tell the turtle to draw a circle. *In the parentheses* `()`, *type in a number to represent the RADIUS of your circle. Tip: The* ***radius*** *of a circle is the distance from the middle of a circle and the edge of the circle.*

```
turtle.circle(85)
```

    a. Draw only part of a circle. *In the parentheses* `()`, *the FIRST number represents the RADIUS of your circle. The SECOND number represents the EXTENT of your circle. This EXTENT is the angle (in degrees) that determines which part of the circle is drawn. Tip: a full circle is 360 degrees.*

```
turtle.circle(85,90)
```

2. Tell the turtle to draw a dot. *In the parentheses* `()`, *type in a number to represent the DIAMETER of your dot. Tip: The* ***diameter*** *of a circle is the distance from one side of a circle to the other.*

```
turtle.dot(15)
```

3. Tell the turtle to hold down or pick up the pen that it is "holding." *When the pen is down, the turtle will draw when it moves. When the pen is up, the turtle will not draw when it moves.*

```
turtle.pendown()
```

```
turtle.penup()
```

4. Change the thickness of the line that the turtle draws. *In the parentheses* `()`, *type in any positive number to set the line width.*

```
turtle.pensize(8)
```

5. Change the shape of the turtle itself! *In the parentheses* `()`, *type in one of the following words in quotes: "****arrow****", "****turtle****", "****circle****", "****square****", "****triangle****", or "****classic****".*

```
turtle.shape("turtle")
```

6. Make the turtle visible or invisible.

```
turtle.hideturtle()
```

```
turtle.showturtle()
```

7. Tell the turtle to go to a certain coordinate position. *In the parentheses* `()`, *the first number represents the value of the X coordinate, while the second number represents the value of the Y coordinate.*

```
turtle.goto(60,30)
```

8. Change either the turtle's X or Y coordinate, but leave the other coordinate the same. *In the parentheses* `()`, *type in the value of the X or Y coordinate you'd like the turtle to move to.*

```
turtle.setx(20)
```

```
turtle.sety(-30)
```

9. Change the color/image of the background. *In the parentheses `()`, type in the name of the color in quotes " " or the name of the image.*

```
turtle.bgcolor("orange")
```

```
turtle.bgpic("testimage.png")
```

10. Erase the lines that the turtle has drawn, but don't move the turtle itself.

```
turtle.clearscreen()
```

11. Erase the lines that the turtle has drawn, and move the turtle back to the origin.

```
turtle.resetscreen()
```

Turtle Practice Problems!

Want to practice working with the turtle? Try to create a program that uses the turtle to draw out some of the following pictures!