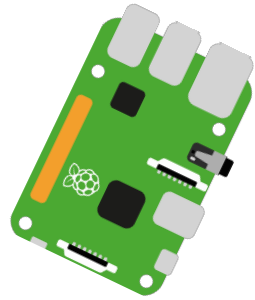


FIREWORKS WITH PYTHON!



In this project we will use the Python Turtle library to create a drawing of a fireworks display. We will randomise the colour, appearance and location of our fireworks too.

Basic Shapes with Turtle



1

Before we start to draw our firework star bursts, it will be useful to understand how we can draw basic shapes with the Python Turtle. Open the "Thonny Python IDE" from the "Programming" menu and enter the following block of code to draw a

simple five pointed star. The lines starting with the hash symbol are comments to explain what each step is doing.

This should draw a simple five pointed star when we run it; however we can do this with far fewer lines of code!

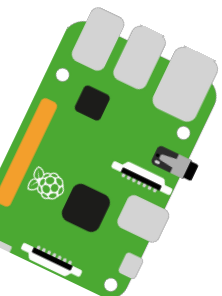
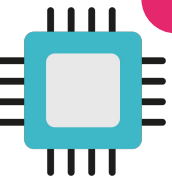
We are going to refactor this code to use a "for loop" to repeat the drawing of each side...

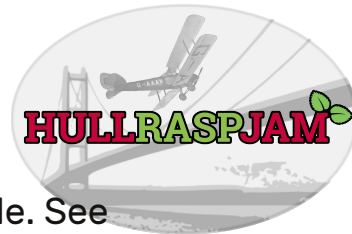
```
# Import our libraries first
import turtle
```

```
# Create our turtle object and set it's shape and speed
fred = turtle.Turtle()
fred.shape("turtle")
fred.speed(0)
```

```
# We can use forward, backward, left and right to move
# our turtle drawing lines as we go
# We can also use penup() and pendown() to lift / lower
# the pen to our canvas
# pencolor("Colour Name") or pencolor([R,G,B]) we also
# come in useful later on!
```

```
fred.forward(100)
fred.backward(100)
fred.right(72)
fred.forward(100)
fred.backward(100)
fred.right(72)
fred.forward(100)
fred.backward(100)
fred.right(72)
fred.forward(100)
fred.backward(100)
fred.right(72)
fred.forward(100)
fred.backward(100)
fred.right(72)
```





Edit the program that you just wrote to match this code. See how we are using a "for loop" to repeat the section of our program that draws each side?

The three lines which come after the "for loop" are indented to show that they are the steps which are iterated inside our loop.

Now see if you can adjust your program to draw a six pointed star instead of a five pointed star...

```
# Import our libraries first
import turtle
```

```
# Create our turtle object and set it's shape and speed
fred = turtle.Turtle()
fred.shape("turtle")
fred.speed(0)
```

```
# We can use forward, backward, left and right to move
# our turtle drawing lines as we go
# We can also use penup() and pendown() to lift / lower
# the pen to our canvas
# pencolor("Colour Name") or pencolor([R,G,B]) we also
# come in useful later on!
```

```
for i in range(5):
    fred.forward(100)
    fred.backward(100)
    fred.right(72)
```

Setting Colours

2

We are now going to look at setting the colour of our turtle. As the comments in the program say, we can either use a colour name or define a value for Red, Green and Blue (R,G,B). We will start with colour names.

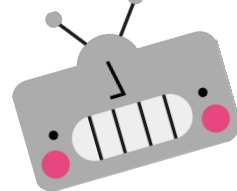
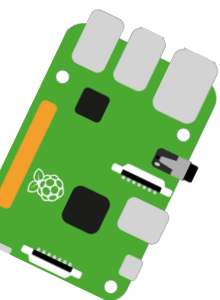
Edit your program, adding this line before your for loop...

Try changing the colour name and see what other colours you can use. Can you get it to work using the RGB method?

Hint: RGB needs to be a list (use square brackets!) of three integers between 0 and 255!

```
fred.pencolor("Blue")
```

```
for i in range(5):
    fred.forward(100)
    fred.backward(100)
    fred.right(72)
```





Moving and Random

3

If we want to move our turtle to draw a new shape, we can use the `penup` and `pendown` functions. Edit your program so that it draws two shapes.

The "**`goto()`**" function takes two integers; an x and y value to determine where we move the turtle to.

We are now also going to introduce some randomisation into our program so that it is a bit different everytime we run it. There are two functions in the random library which will be useful for us, **`randint`** and **`choice`**.

We can now use **`choice`** to randomly select a colour name

```
fred.pencolor("Blue")
```

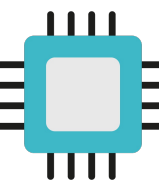
```
for i in range(5):  
    fred.forward(100)  
    fred.backward(100)  
    fred.right(72)
```

```
fred.penup()  
fred.goto(100, 150)  
fred.pendown()
```

```
fred.pencolor("Red")
```

```
for i in range(5):  
    fred.forward(100)  
    fred.backward(100)  
    fred.right(72)
```

from a list and we can use **`randint`** to select a random integer from a start and end value. Update your program to match this and run it a few times. You should see a different coloured shape each time. One uses the colour name method and the other uses the RGB method.



```
# Import our libraries first  
import turtle  
from random import randint, choice
```

```
# Create our turtle and set it's shape, speed and colour mode  
turtle.colormode(255)  
fred = turtle.Turtle()  
fred.shape("turtle")  
fred.speed(0)
```

```
colours = ["red", "blue", "green", "yellow", "magenta", "orange"]
```

```
fred.pencolor(choice(colours))  
for i in range(5):  
    fred.forward(100)  
    fred.backward(100)  
    fred.right(72)
```

```
fred.penup()  
fred.goto(100, 150)  
fred.pendown()
```

```
fred.pencolor(randint(0,255),randint(0,255),randint(0,255))  
for i in range(5):  
    fred.forward(100)  
    fred.backward(100)  
    fred.right(72)
```



Functions

4

We use functions in programming when we want to create pieces of our program that we want to use over and over again. We can also pass parameters to our function which allow it to behave slightly differently each time we run it. In the final challenge you are going to create a program which draws firework star bursts randomly across the screen. To do this we will need to create a function for moving to a random point and a function to draw a randomly coloured starburst.

We write a function like this:

```
def myCoolFuntion(parameter1, parameter2):
```

```
    # Do some stuff now!  
    for i in range(5):  
        turtle.forward(parameter1)  
        turtle.backward(parameter1)  
        turtle.right(parameter2)
```

```
myCoolFunction(100, 72)
```

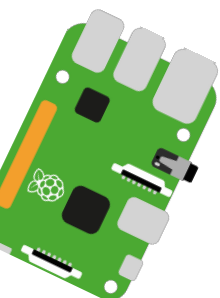
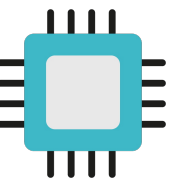
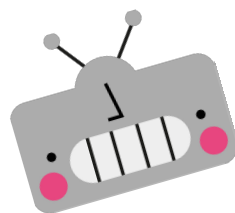
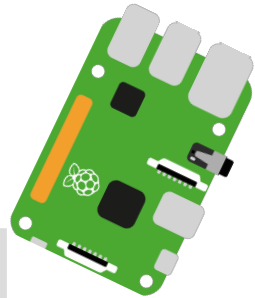
We always start with the "**def**" keyword, followed by the functions name and an open and close parentheses. If we want to pass any parameters into our function they are set inside the parentheses. This line always ends with a colon.

We then write the lines which will be executed when we call our function.

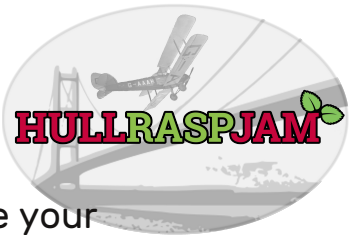
We can then use our function by typing it's name and the parentheses, passing any parameters if required.

The example above will always draw a five pointed star; but it can be changed to draw a star of any number of legs... See if you can work out what needs changing...

Hint: You need to work out the relationship between the number of sides or legs of your star and the number of degrees you turn!



Challenge



5

You are now going to put everything together to create your own fireworks display!

The code block gets you started and sets out what you need to do. Try your best and see what you come up with. There is an example solution but you will need to admit defeat and ask for it!

```
# Draw a fireworks display, using Python's random module
# to randomise the colour, appearance, and location of the
# individual starbursts.
```

```
import turtle
from random import randint, choice
```

```
# Use a black background
screen = turtle.Screen()
screen.bgcolor("black")
```

```
# Create our turtle and set it's shape, speed and colour mode
turtle.colormode(255)
fred = turtle.Turtle()
fred.shape("turtle")
fred.speed(0)
```

```
# List of colors available for the fireworks
# or use random RGB values
colours = ["blue", "magenta", "red", "green", "orange", "yellow"]
```

```
# Step 1: Define a function drawStarburst, which draws an
# individual starburst. It should have a random number of
# legs, a random length for each leg and a random colour
```

```
# Step 2: Define a function moveRandomly, which moves the
# turtle to a random location on the screen without drawing
# anything.
```

```
# Step 3: Use the drawStarburst and moveRandomly functions
# to draw a bunch of starbursts in different locations on
# the screen.
```

