Learn to Program with Turtle!

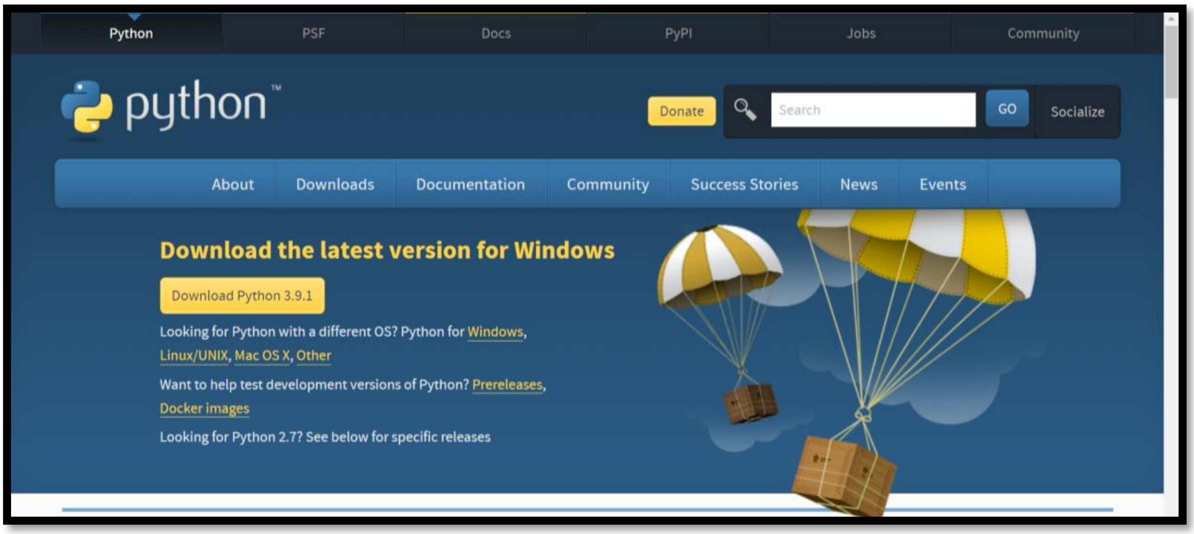
By Reuben Mitchell

Table of Contents

Getting Started with Python

Downloading Python

Python is completely free and can be downloaded from https://www.python.org/downloads/.



Download the latest version for Windows

Looking for Python with a different OS? Python for Windows, Linux/UNIX, Mac OS X, Other

Want to help test development versions of Python? Prereleases, Docker images

Looking for Python 2.7? See below for specific releases

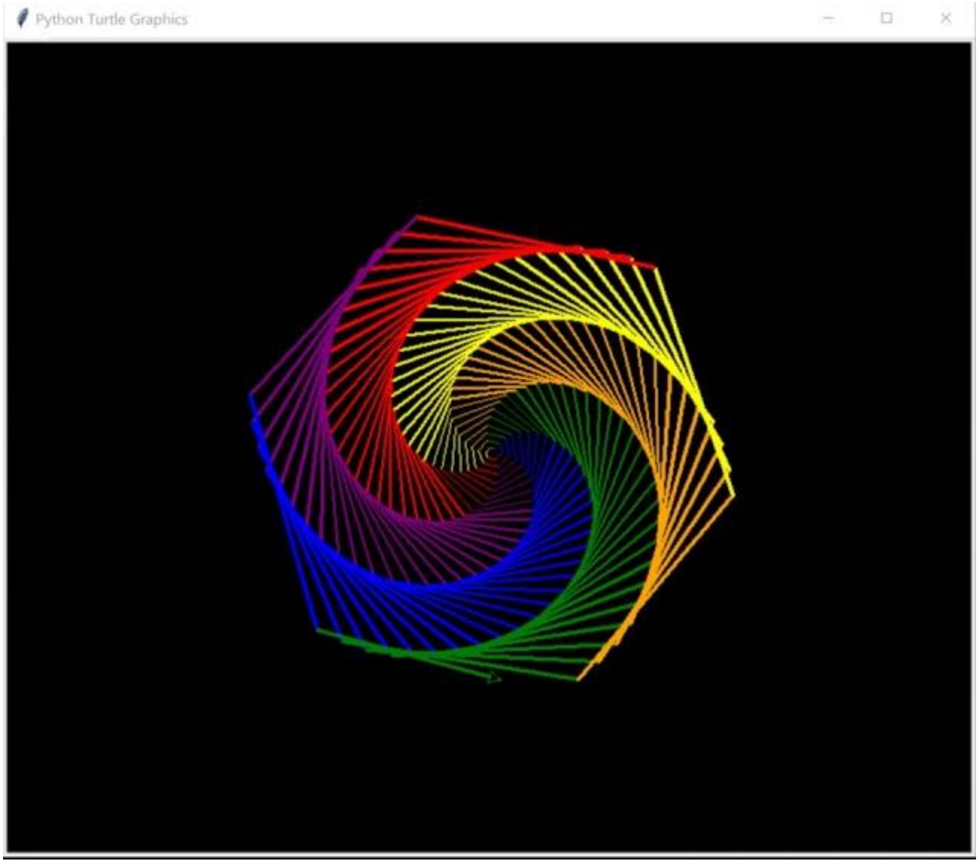
Writing Your First Program

If you are new to Python or programming and want to try writing your first program before learning how to use Turtle, then this tutorial is perfect for you: https://www.pitt.edu/~naraehan/python3/getting_started_win_first_try.html.

Introduction to Turtle

What is Turtle?

Turtle is a Python programming tool that can be used to draw pictures like this one:

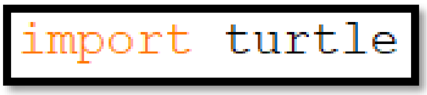


How to use Turtle

Turtle is what's called a "Python module".

For you to be able to use it, you must import the module.

We do this by typing "import turtle" at the start of all our programs.



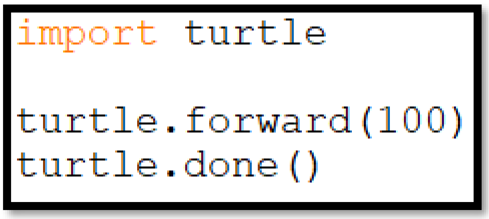
```
import turtle
```

Making Your First Turtle Program

Now that you have imported turtle, you can start to use it.

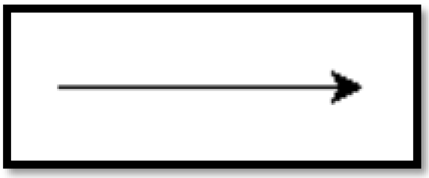
The simplest program you can make is to draw a straight line.

Type the following code into a Python file and run it.



```
import turtle

turtle.forward(100)
turtle.done()
```

If successful, a new window will open, and you will see the following drawing.



Congratulations! You've just made your first turtle program!

Code Explanation

turtle.forward(100) ➔ Tells Python to draw a line of length 100 (no units).

turtle.done() ➔ Tells Python you have finished your drawing (you should add this to the end of all of your programs).

More Common Commands

Don’t worry if the following table doesn’t make much sense yet. There will be example code later which aims to make everything clearer.

Command	Description
turtle.forward(x)	Draw a line in the forward direction of size x (where x is a number)
turtle.backward(x)	Draw a line in the backward direction of size x (where x is a number)
turtle.right(x)	Turn the pointer to the right by x degrees (where x is a number)
turtle.left(x)	Turn the pointer to the left by x degrees (where x is a number)
turtle.circle(x)	Draw a circle with a radius of x (where x is a number)
turtle.pendown()	When the pen is down it will draw when moving
turtle.penup()	When the pen is up it will move without drawing
turtle.pensize(x)	Sets the width of the pen to be x wide. The larger x is, the thicker the pen will be (where x is a positive number)
turtle.pencolor(x)	Sets the colour of the line when it is drawing (where x is the name of a colour)
turtle.fillcolor(x)	Sets the colour of the inside of the shape when drawing (where x is the name of the colour)
turtle.begin_fill()	Tells the turtle when to start filling in the shape with the specified colour (to be used just before drawing a shape to be filled)
turtle.end_fill()	Tells the turtle when to stop filling in the shape with the specified colour (to be called just after drawing a shape to be filled)
turtle.showturtle()	Show the pointer when drawing

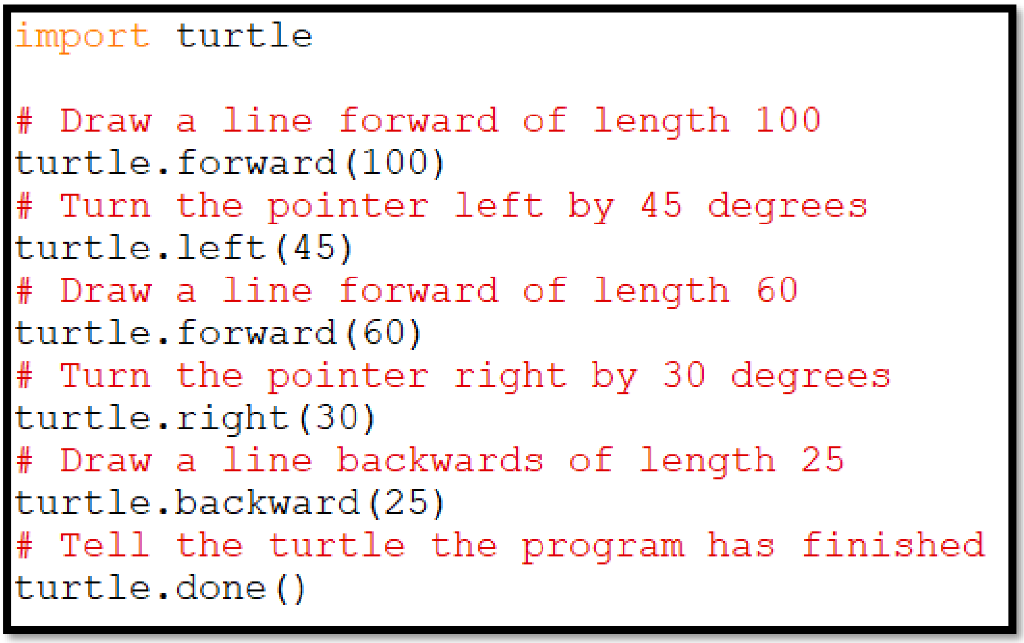
turtle.hideturtle()	Don’t show the pointer when drawing
turtle.shape(x)	Change the shape of the pointer. Try setting x to “turtle”. It’s the best one!
turtle.bgcolor(x)	Sets the background colour of the window created (where x is the name of a colour)
turtle.done()	To be called at the end of your program

For a complete list of all of the commands available visit: https://docs.python.org/3/library/turtle.html#:~:text=Turtle%20graphics%20is%20a%20popular%20way%20for%20introducing%20programming%20to%20kids.&text=The%20turtle%20module%20is%20an,)%20100%25%20compatible%20with%20it.

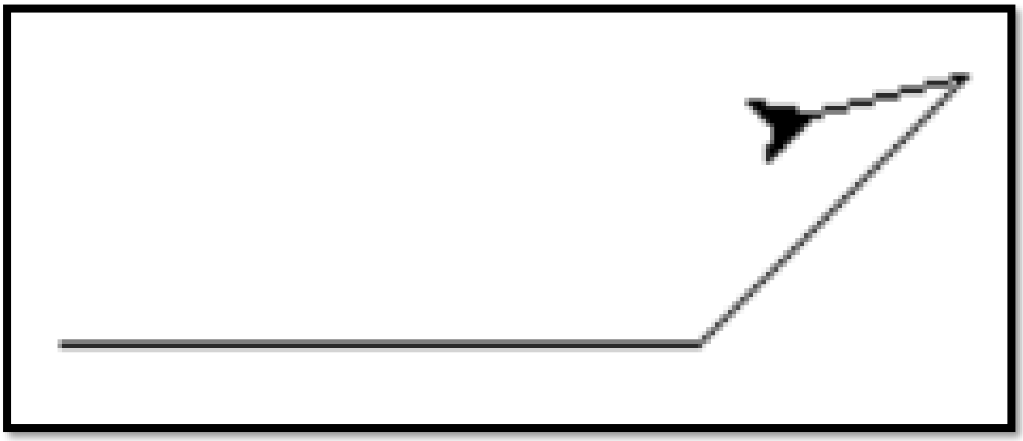
Example Code

It's worth noting that any text that is in red is a comment. A comment is used to explain what is happening in your program. To write a comment, start the line with a hashtag "#".

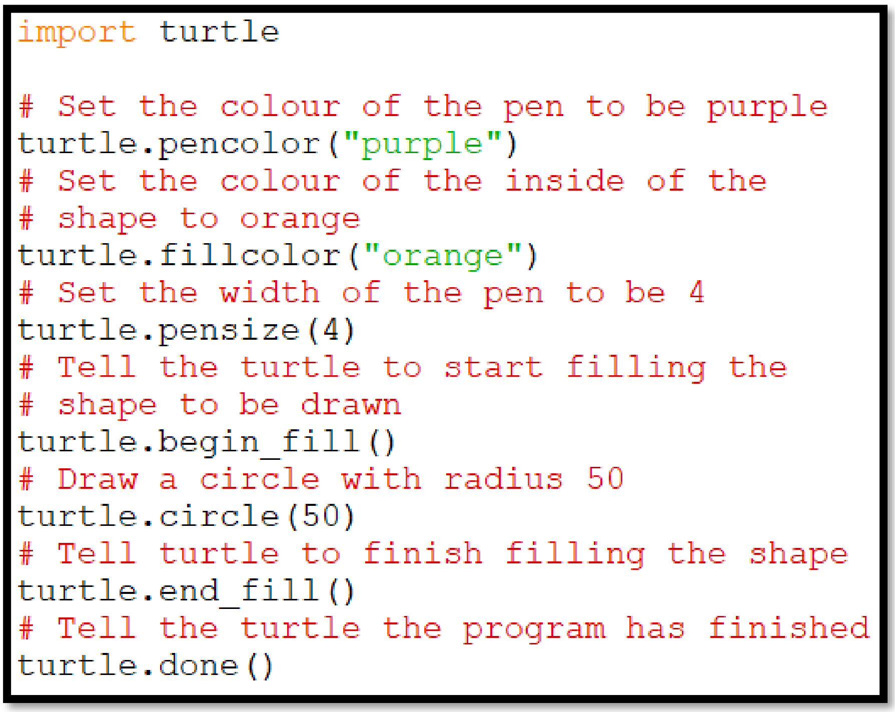
Example Code 1



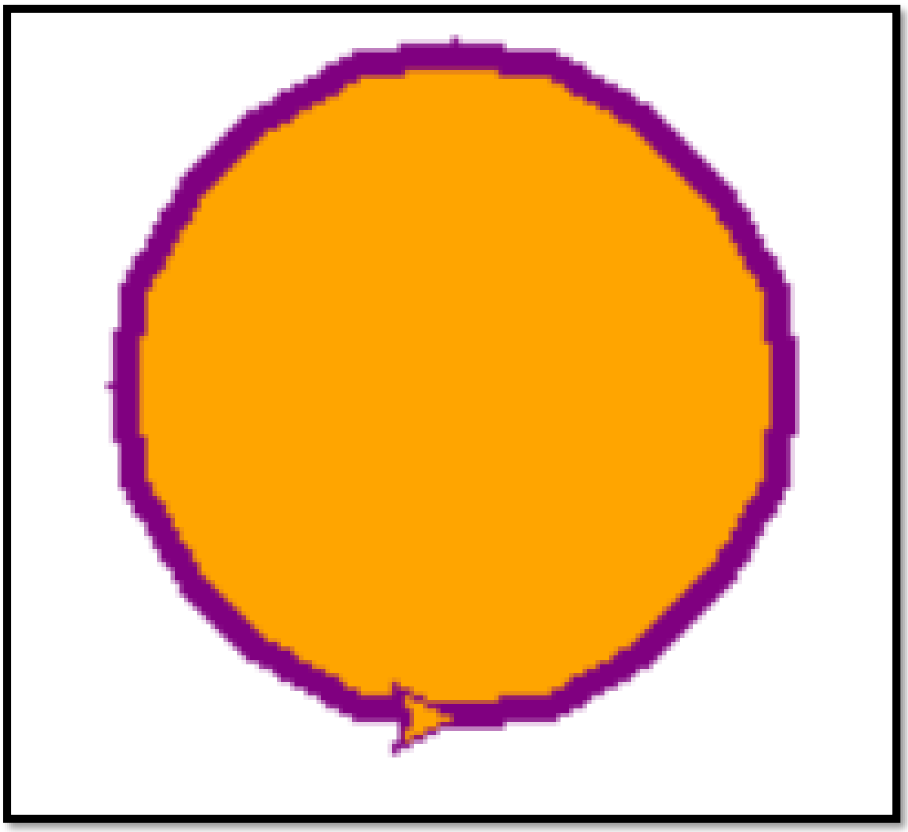
```
import turtle

# Draw a line forward of length 100
turtle.forward(100)
# Turn the pointer left by 45 degrees
turtle.left(45)
# Draw a line forward of length 60
turtle.forward(60)
# Turn the pointer right by 30 degrees
turtle.right(30)
# Draw a line backwards of length 25
turtle.backward(25)
# Tell the turtle the program has finished
turtle.done()
```



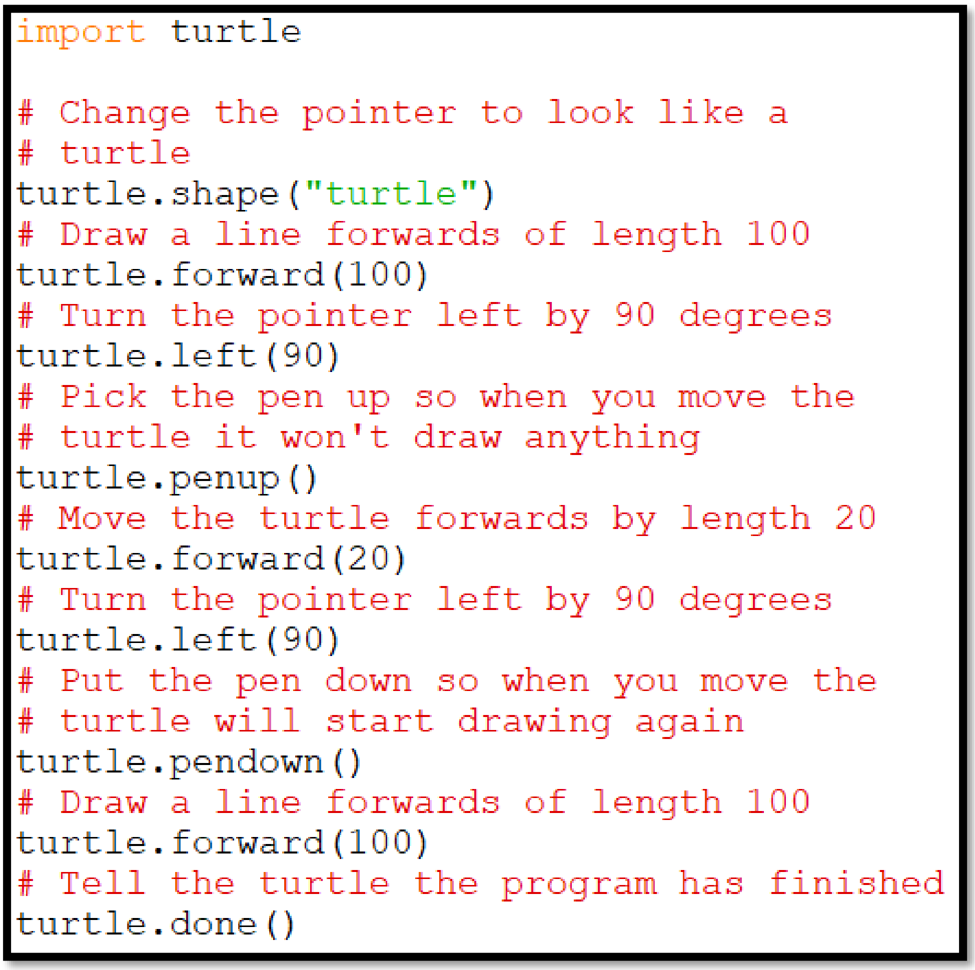
Example Code 2



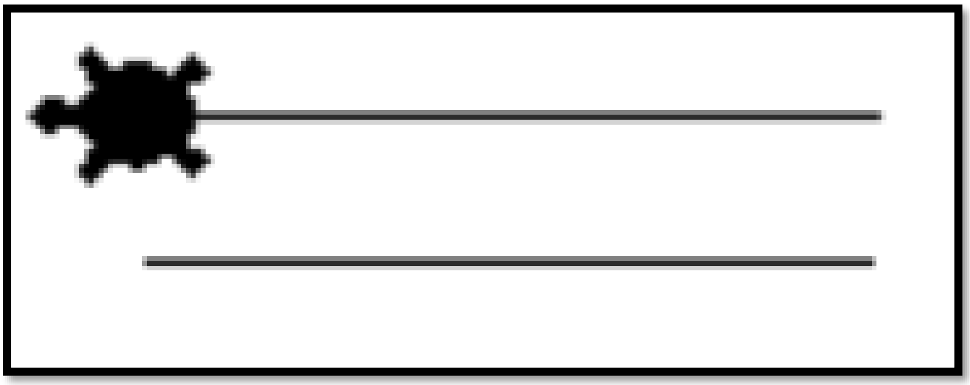
```
import turtle

# Set the colour of the pen to be purple
turtle.pencolor("purple")
# Set the colour of the inside of the
# shape to orange
turtle.fillcolor("orange")
# Set the width of the pen to be 4
turtle.pensize(4)
# Tell the turtle to start filling the
# shape to be drawn
turtle.begin_fill()
# Draw a circle with radius 50
turtle.circle(50)
# Tell turtle to finish filling the shape
turtle.end_fill()
# Tell the turtle the program has finished
turtle.done()
```



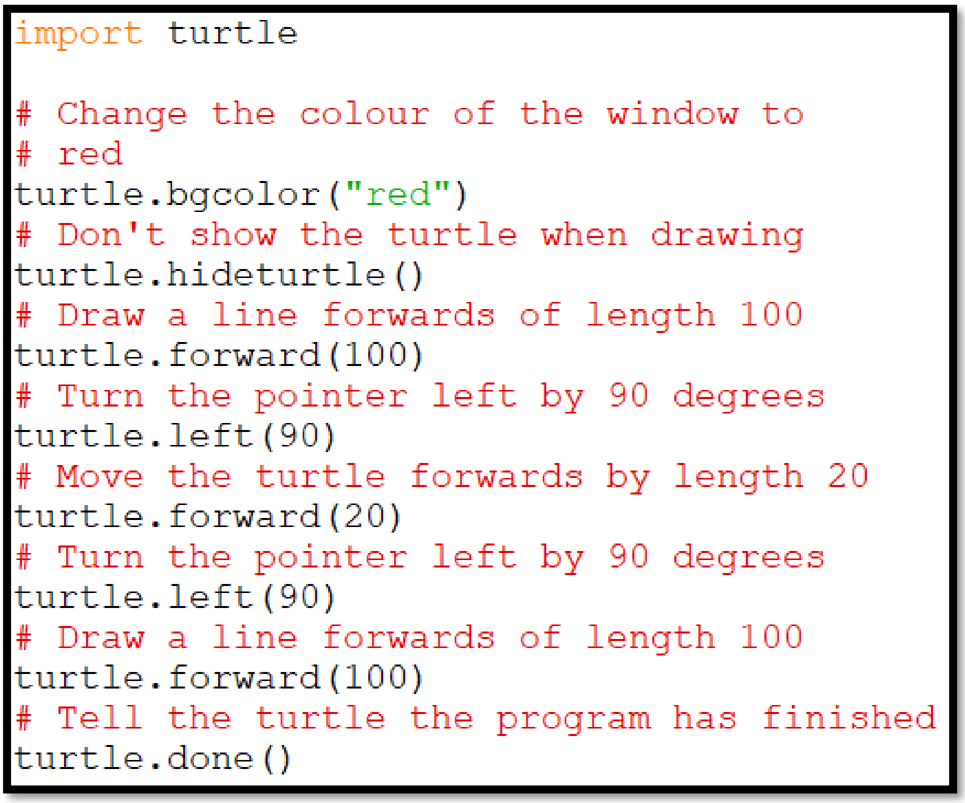
Example Code 3



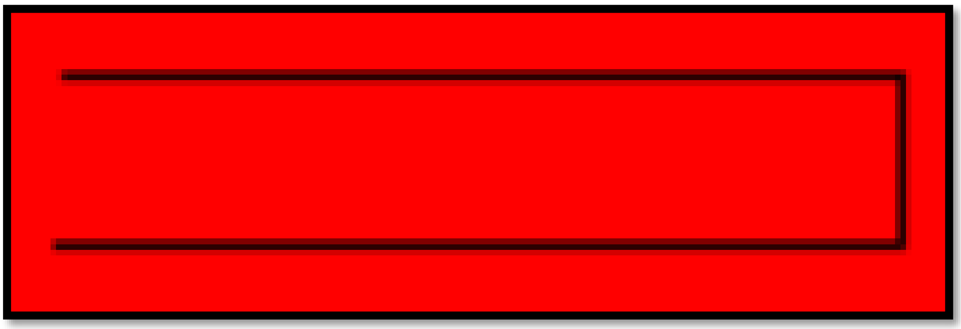
```
import turtle

# Change the pointer to look like a
# turtle
turtle.shape("turtle")
# Draw a line forwards of length 100
turtle.forward(100)
# Turn the pointer left by 90 degrees
turtle.left(90)
# Pick the pen up so when you move the
# turtle it won't draw anything
turtle.penup()
# Move the turtle forwards by length 20
turtle.forward(20)
# Turn the pointer left by 90 degrees
turtle.left(90)
# Put the pen down so when you move the
# turtle will start drawing again
turtle.pendown()
# Draw a line forwards of length 100
turtle.forward(100)
# Tell the turtle the program has finished
turtle.done()
```



Example Code 4



```
import turtle

# Change the colour of the window to
# red
turtle.bgcolor("red")
# Don't show the turtle when drawing
turtle.hideturtle()
# Draw a line forwards of length 100
turtle.forward(100)
# Turn the pointer left by 90 degrees
turtle.left(90)
# Move the turtle forwards by length 20
turtle.forward(20)
# Turn the pointer left by 90 degrees
turtle.left(90)
# Draw a line forwards of length 100
turtle.forward(100)
# Tell the turtle the program has finished
turtle.done()
```



Introduction to Main Content

Outline

Throughout the next section of this tutorial, you will learn how to draw a:

1. Square
2. Triangle
3. Pentagon
4. Hexagon
5. Octagon
6. Rectangle

Each shape will be split into 3 sections:

1. Step by Step ➔ This will explain the theory behind the drawing of the shape.
2. Output ➔ This will show you an image of the shape you will learn to draw.
3. Programming ➔ This will include all the code you need to draw the shape. This section will be split into 3 different levels. Beginner, Intermediate and Advanced.

Beginner Code

The beginner code is a great place to start if you have never programmed before. Once you have mastered the beginner code you can move onto the intermediate code.

Intermediate Code

The intermediate code introduces the concept of loops. In particular, a type of loop called a "while loop". Loops are a fundamental part of programming because they allow us to run the same section of code repeatedly without the need to write long lists of commands. If you want to learn more about loops, the following is a great resource: https://www.learnpython.org/en/Loops.

Advanced Code

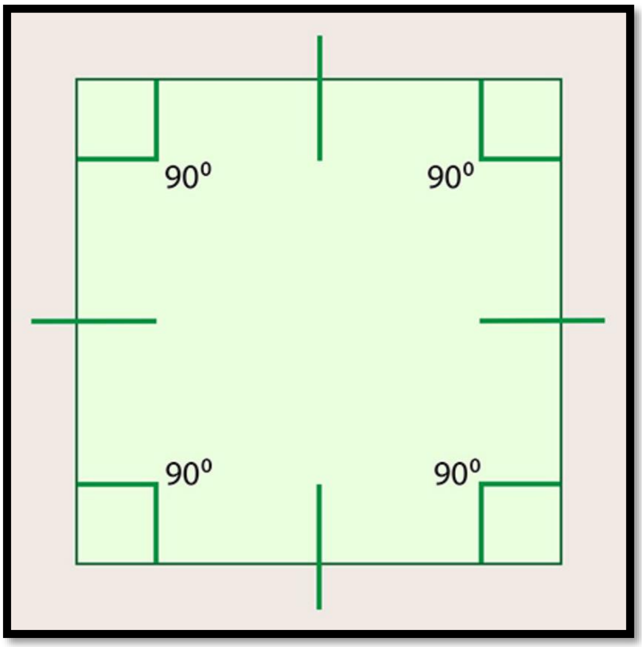
The advanced code uses slightly more complex loops and introduces the concept of functions. Functions are also a fundamental part of programming because they allow us re-use sections of our code. If you want to learn more about functions, the following is a great resource: https://www.learnpython.org/en/Functions.

How to Draw a Square

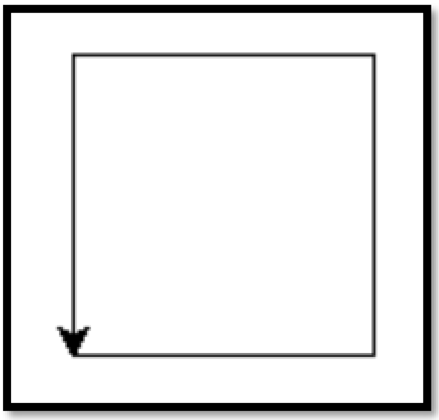
Step by Step:

Starting from the bottom left corner:

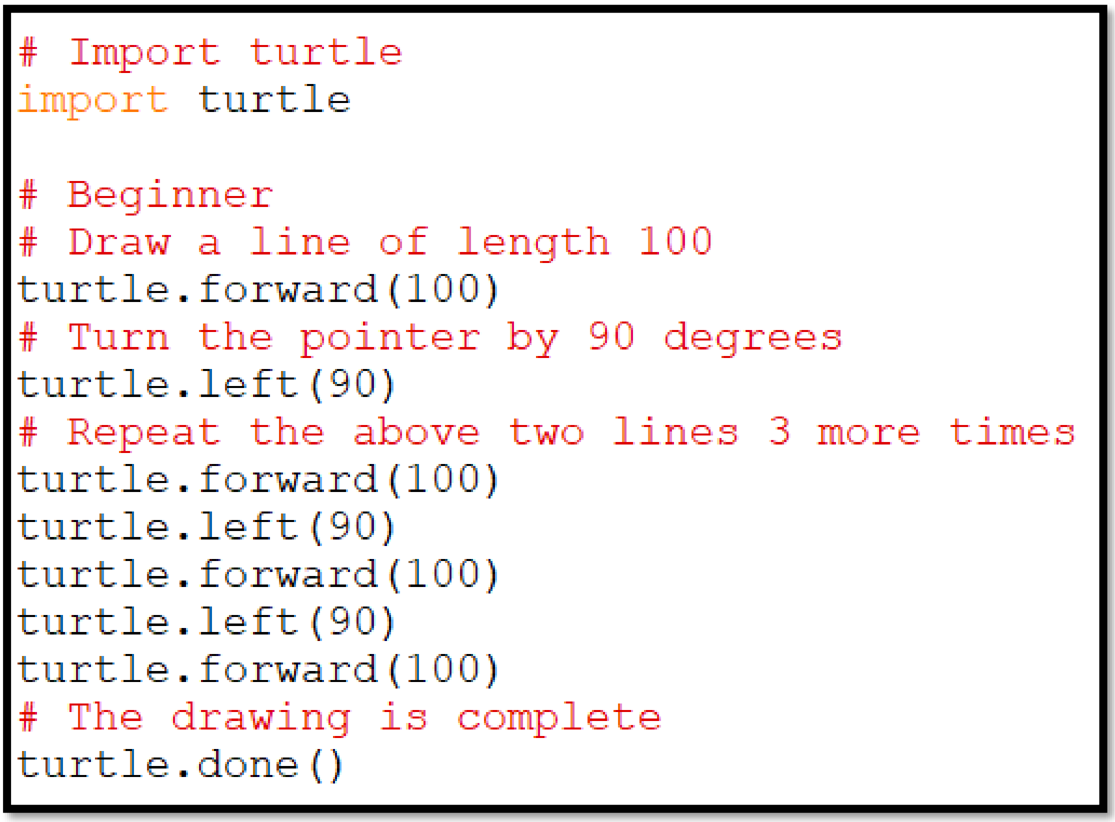
1. Draw a straight line of length 100.
2. Rotate the pointer by 90 degrees.
3. Repeat steps 1-2 three more times.



Output:

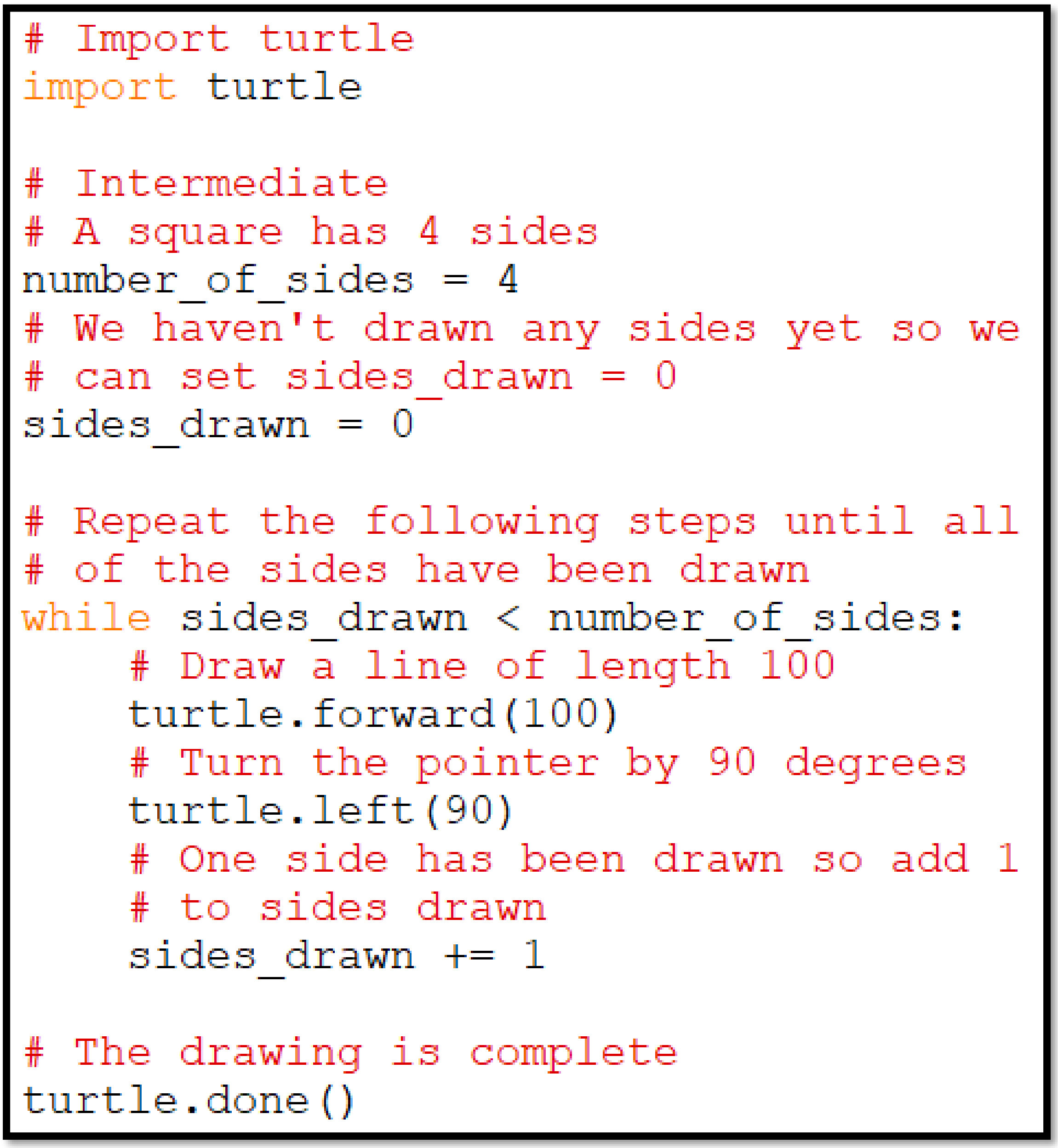


Beginner:



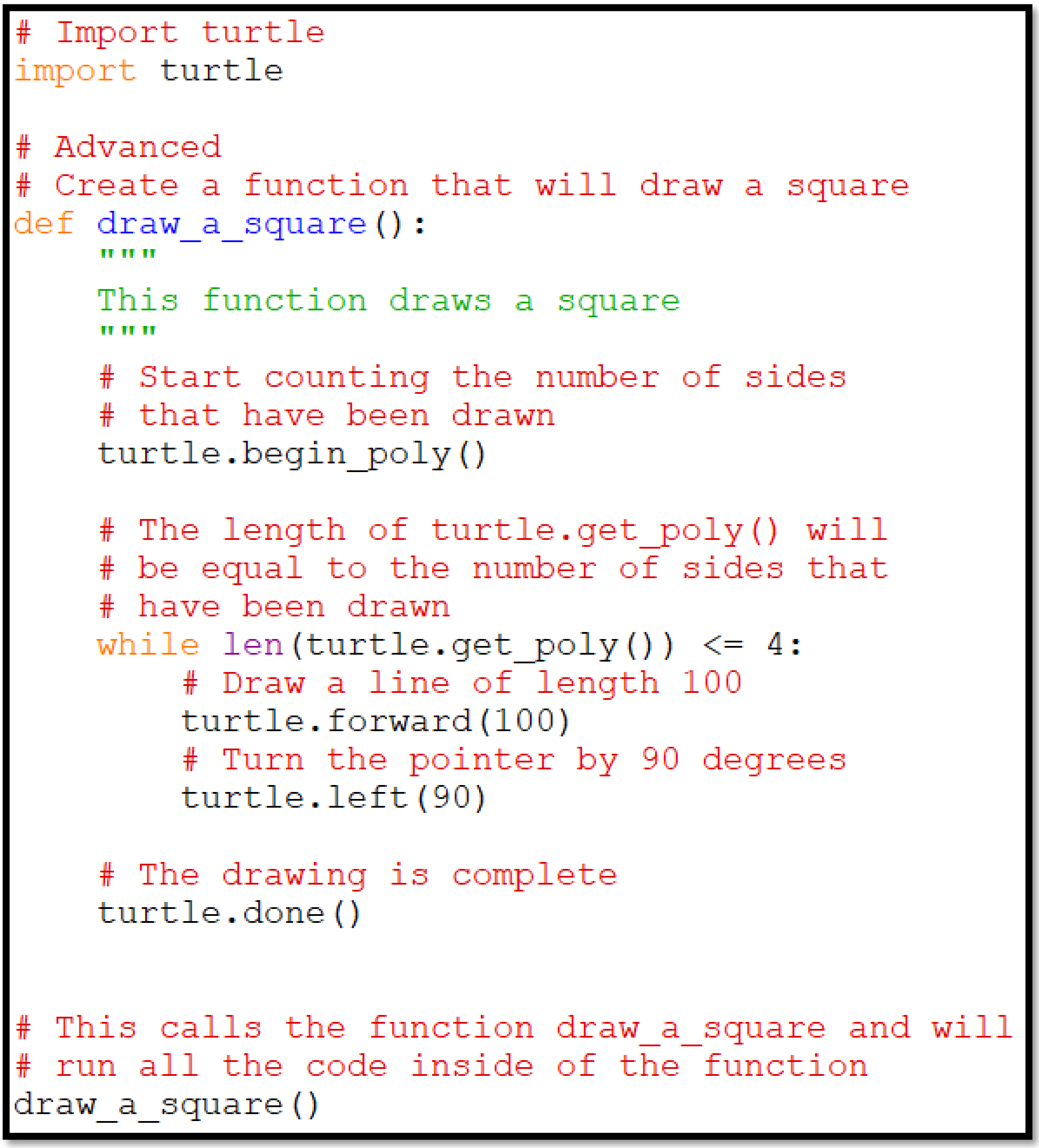
```
# Import turtle
import turtle

# Beginner
# Draw a line of length 100
turtle.forward(100)
# Turn the pointer by 90 degrees
turtle.left(90)
# Repeat the above two lines 3 more times
turtle.forward(100)
turtle.left(90)
turtle.forward(100)
turtle.left(90)
turtle.forward(100)
# The drawing is complete
turtle.done()
```

Intermediate:



```
# Import turtle
import turtle

# Intermediate
# A square has 4 sides
number_of_sides = 4
# We haven't drawn any sides yet so we
# can set sides_drawn = 0
sides_drawn = 0

# Repeat the following steps until all
# of the sides have been drawn
while sides_drawn < number_of_sides:
    # Draw a line of length 100
    turtle.forward(100)
    # Turn the pointer by 90 degrees
    turtle.left(90)
    # One side has been drawn so add 1
    # to sides drawn
    sides_drawn += 1

# The drawing is complete
turtle.done()
```

Advanced



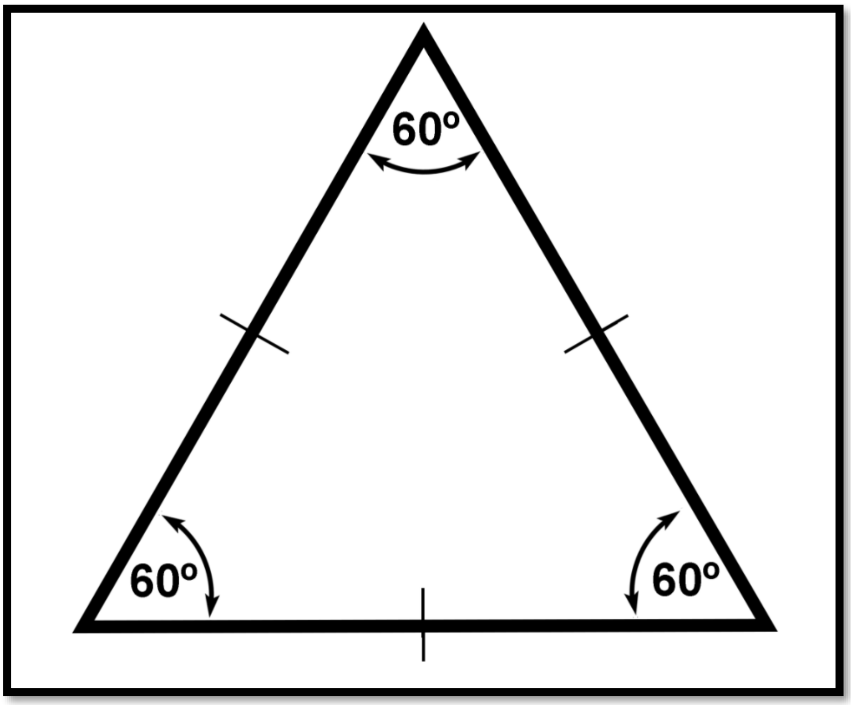
```
# Import turtle
import turtle

# Advanced
# Create a function that will draw a square
def draw_a_square():
    """
    This function draws a square
    """
    # Start counting the number of sides
    # that have been drawn
    turtle.begin_poly()

    # The length of turtle.get_poly() will
    # be equal to the number of sides that
    # have been drawn
    while len(turtle.get_poly()) <= 4:
        # Draw a line of length 100
        turtle.forward(100)
        # Turn the pointer by 90 degrees
        turtle.left(90)

    # The drawing is complete
    turtle.done()


# This calls the function draw_a_square and will
# run all the code inside of the function
draw_a_square()
```

How to Draw a Triangle

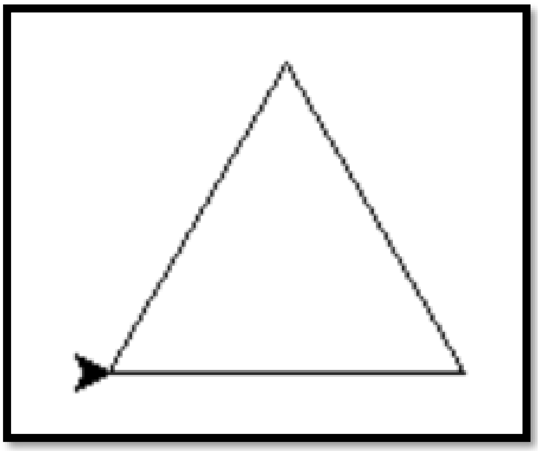
Step by Step:

Starting from the bottom left corner:

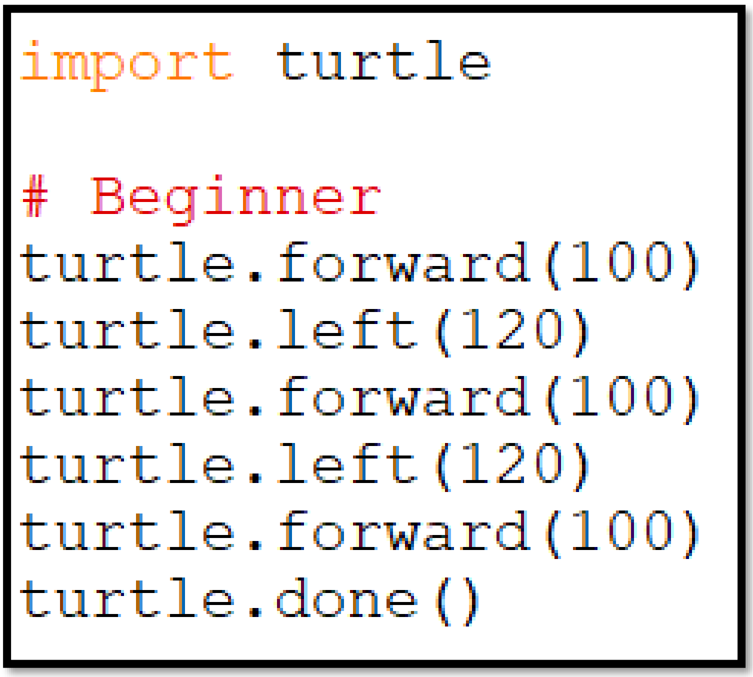
1. Draw a line of length 100.
2. Turn the pointer left by 120 degrees.
3. Repeat steps 1-2 two more times.



Output:

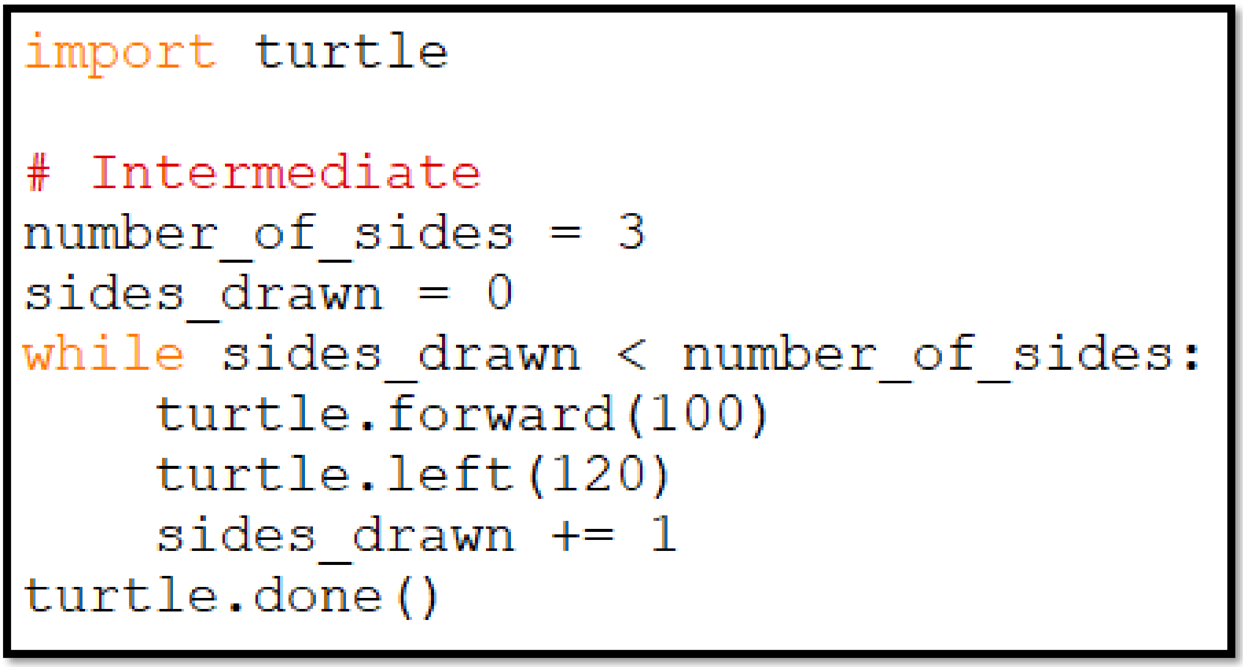


Beginner:



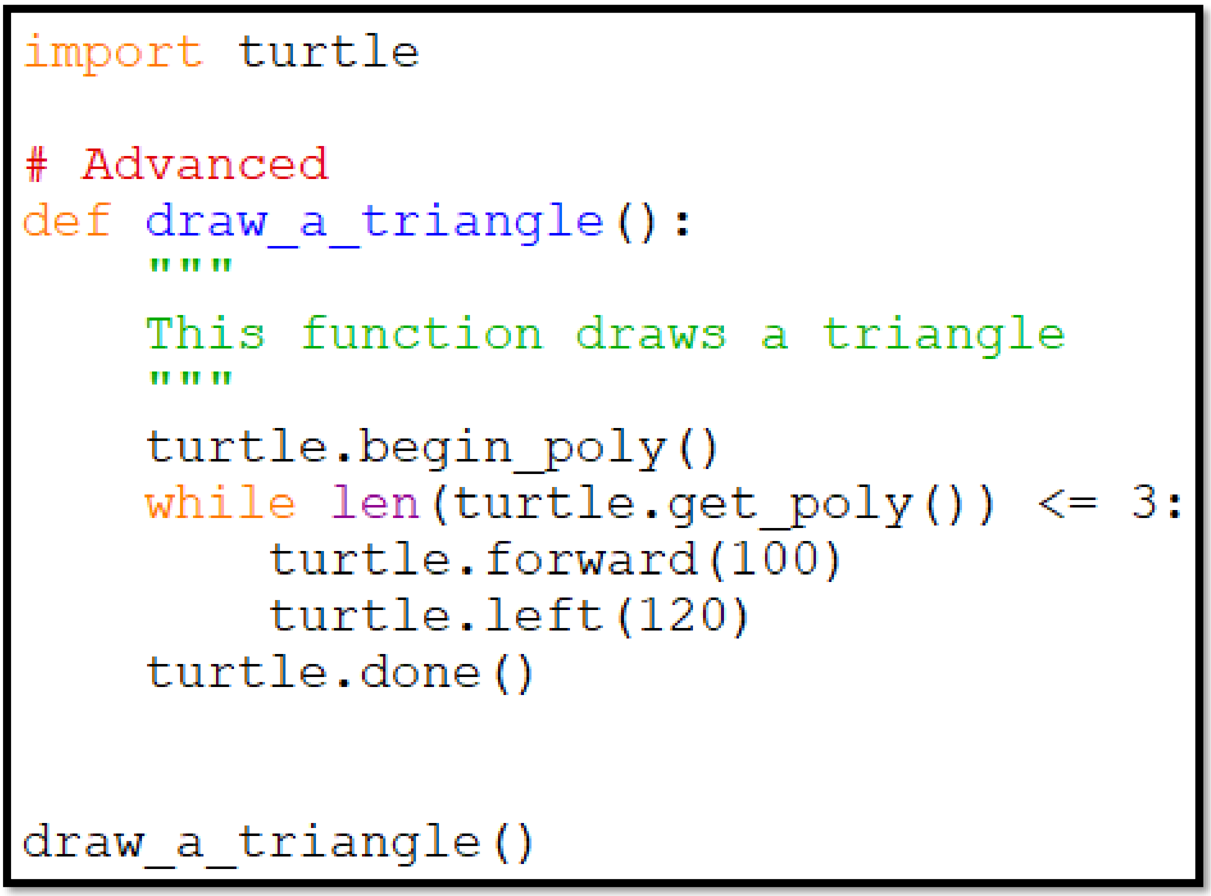
```
import turtle

# Beginner
turtle.forward(100)
turtle.left(120)
turtle.forward(100)
turtle.left(120)
turtle.forward(100)
turtle.done()
```

Intermediate:



```
import turtle

# Intermediate
number_of_sides = 3
sides_drawn = 0
while sides_drawn < number_of_sides:
    turtle.forward(100)
    turtle.left(120)
    sides_drawn += 1
turtle.done()
```

Advanced:



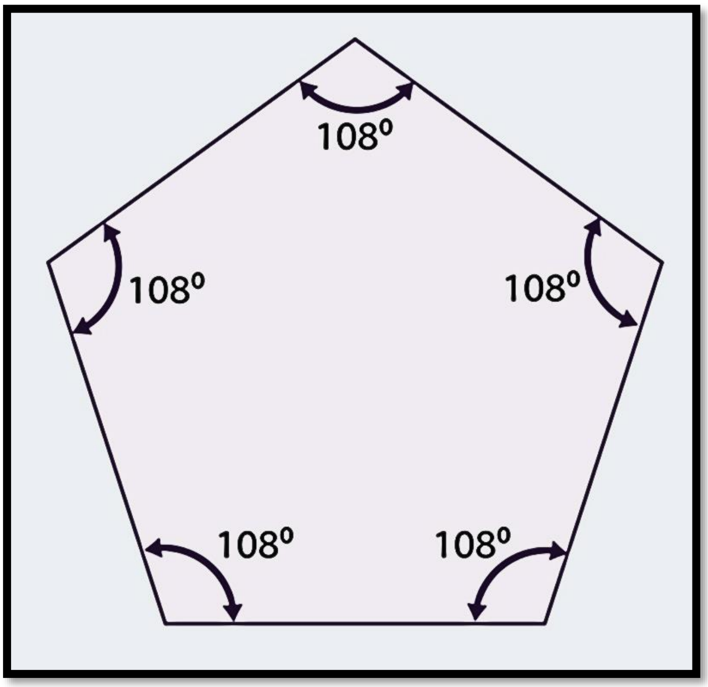
```python
import turtle

# Advanced
def draw_a_triangle():
    """
    This function draws a triangle
    """
    turtle.begin_poly()
    while len(turtle.get_poly()) <= 3:
        turtle.forward(100)
        turtle.left(120)
    turtle.done()


draw_a_triangle()
```

How to Draw a Pentagon

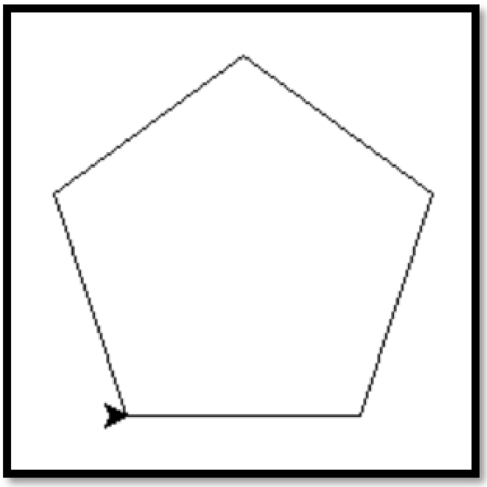
Step by Step

Starting from the bottom left corner:

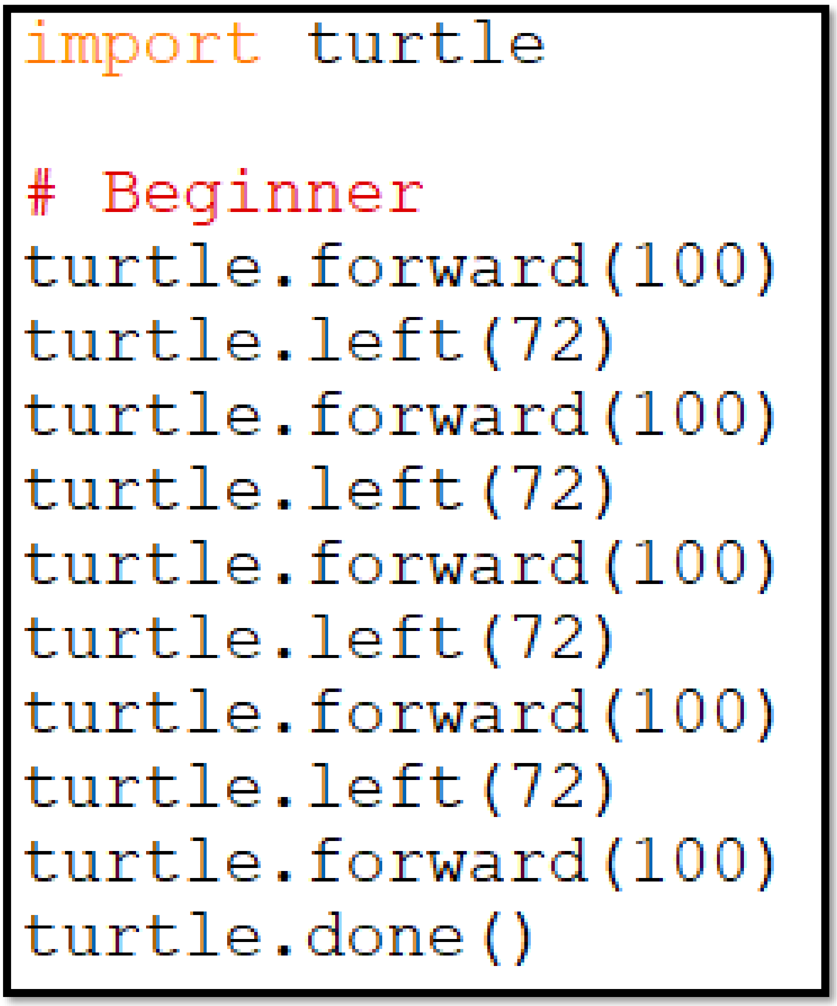
1. Draw a line of length 100.
2. Turn the pointer left by 72 degrees.
3. Repeat steps 1-2 4 more times.



Output:

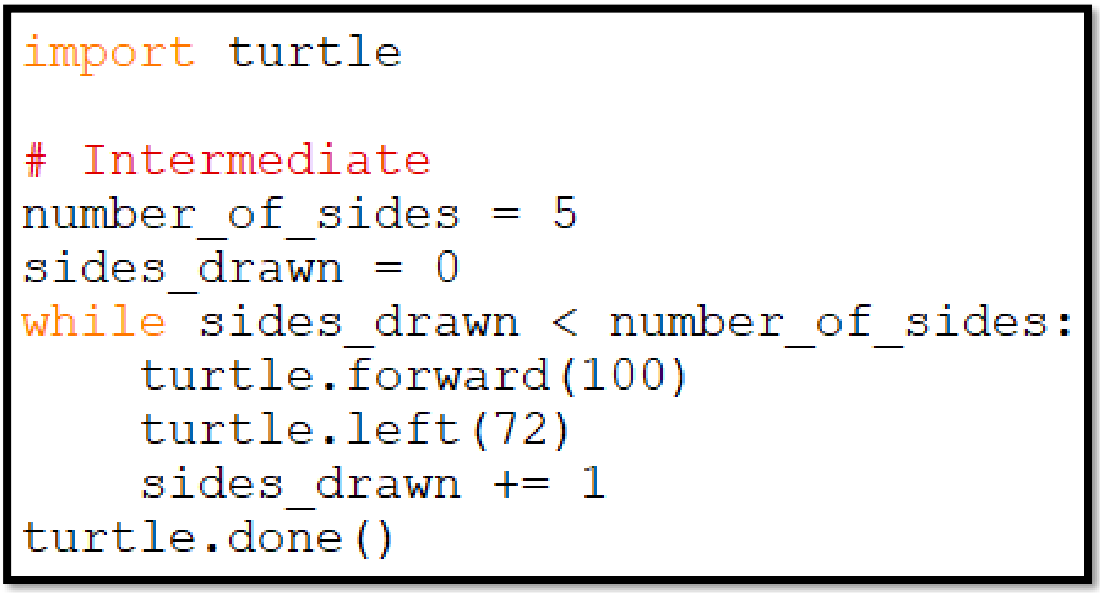


Beginner:



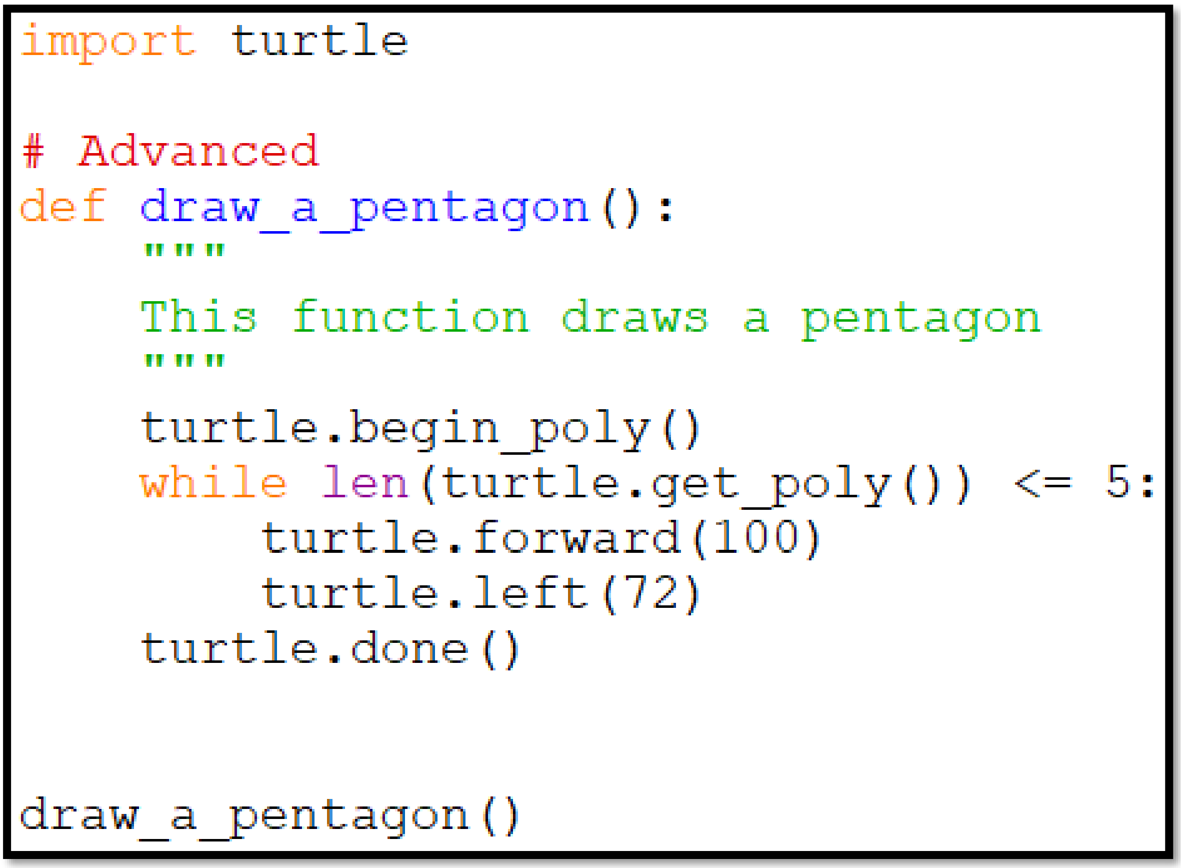
```
import turtle

# Beginner
turtle.forward(100)
turtle.left(72)
turtle.forward(100)
turtle.left(72)
turtle.forward(100)
turtle.left(72)
turtle.forward(100)
turtle.left(72)
turtle.forward(100)
turtle.done()
```

Intermediate:



```python
import turtle

# Intermediate
number_of_sides = 5
sides_drawn = 0
while sides_drawn < number_of_sides:
    turtle.forward(100)
    turtle.left(72)
    sides_drawn += 1
turtle.done()
```

Advanced:



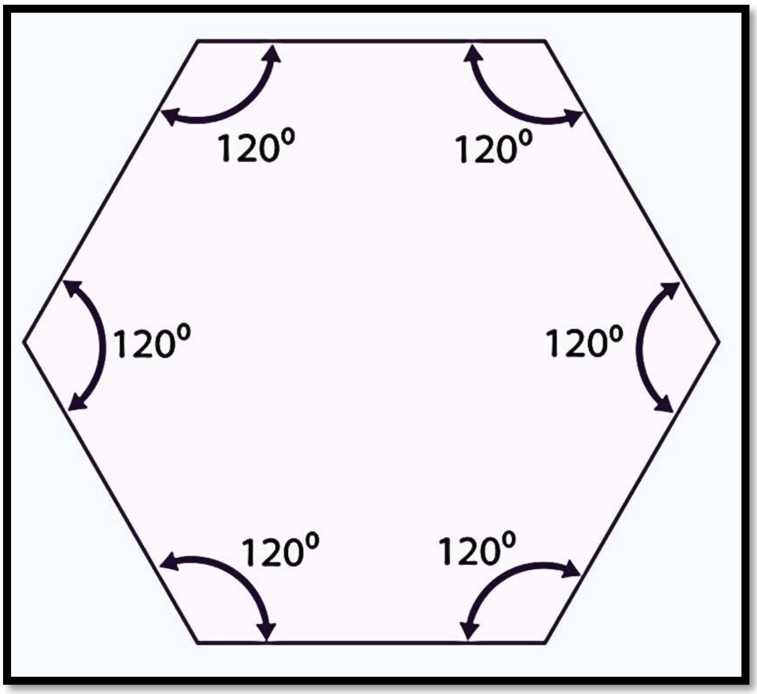
```python
import turtle

# Advanced
def draw_a_pentagon():
    """
    This function draws a pentagon
    """
    turtle.begin_poly()
    while len(turtle.get_poly()) <= 5:
        turtle.forward(100)
        turtle.left(72)
    turtle.done()


draw_a_pentagon()
```

How to Draw a Hexagon

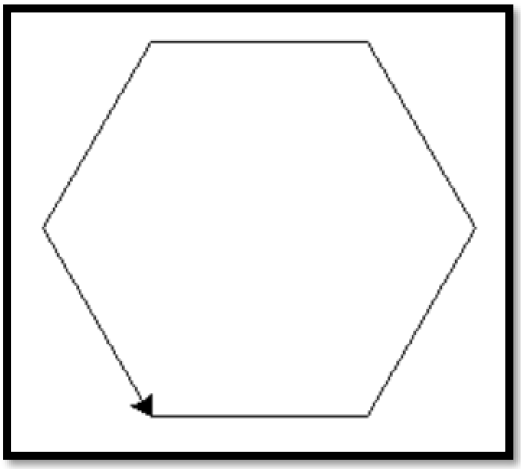
Step by Step

Starting from the bottom left corner:

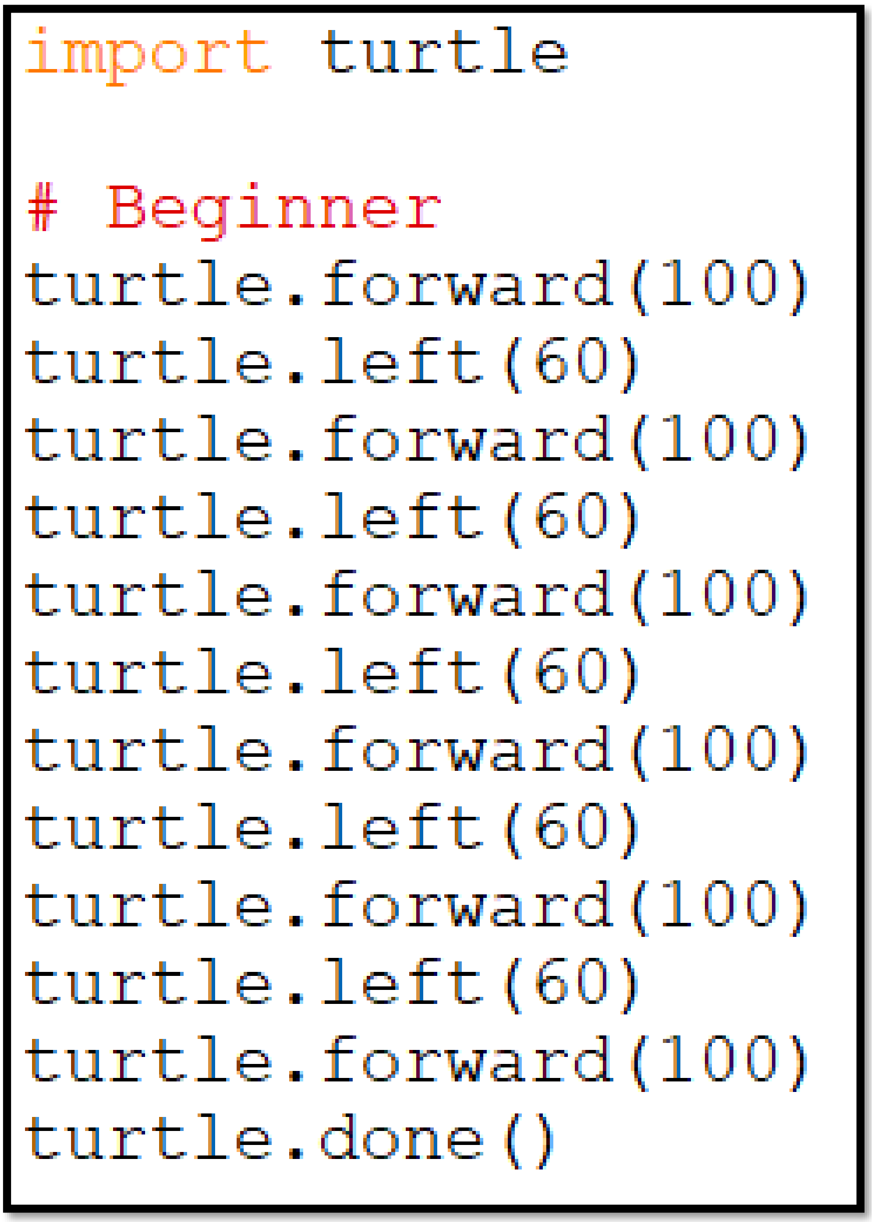
1. Draw a line of length 100.
2. Turn the pointer left by 60 degrees.
3. Repeat steps 1-2 five more times.



Output:

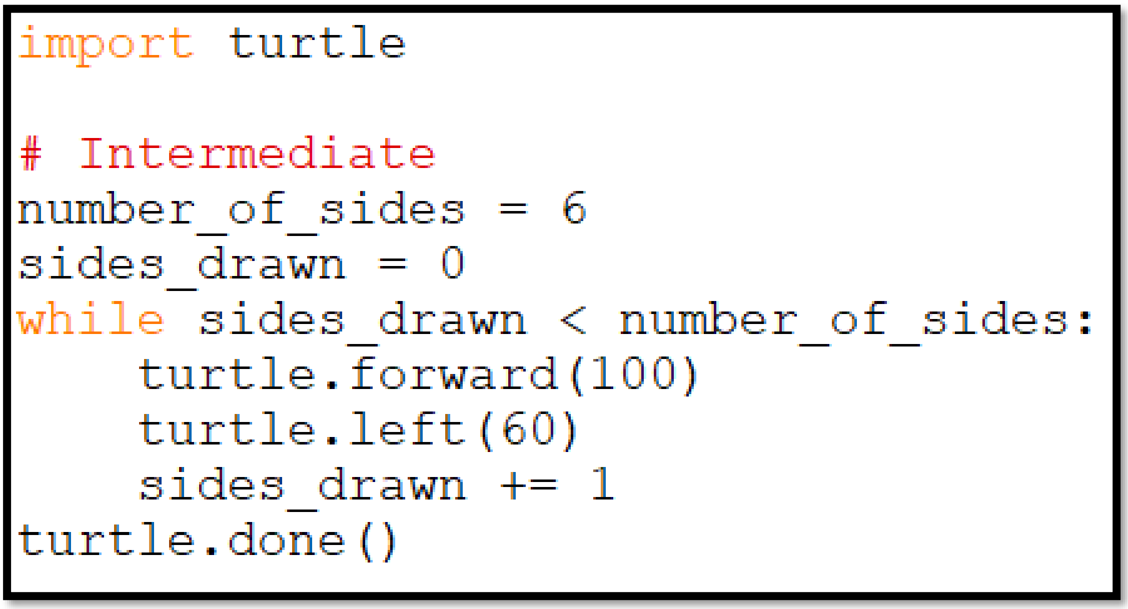


Beginner:



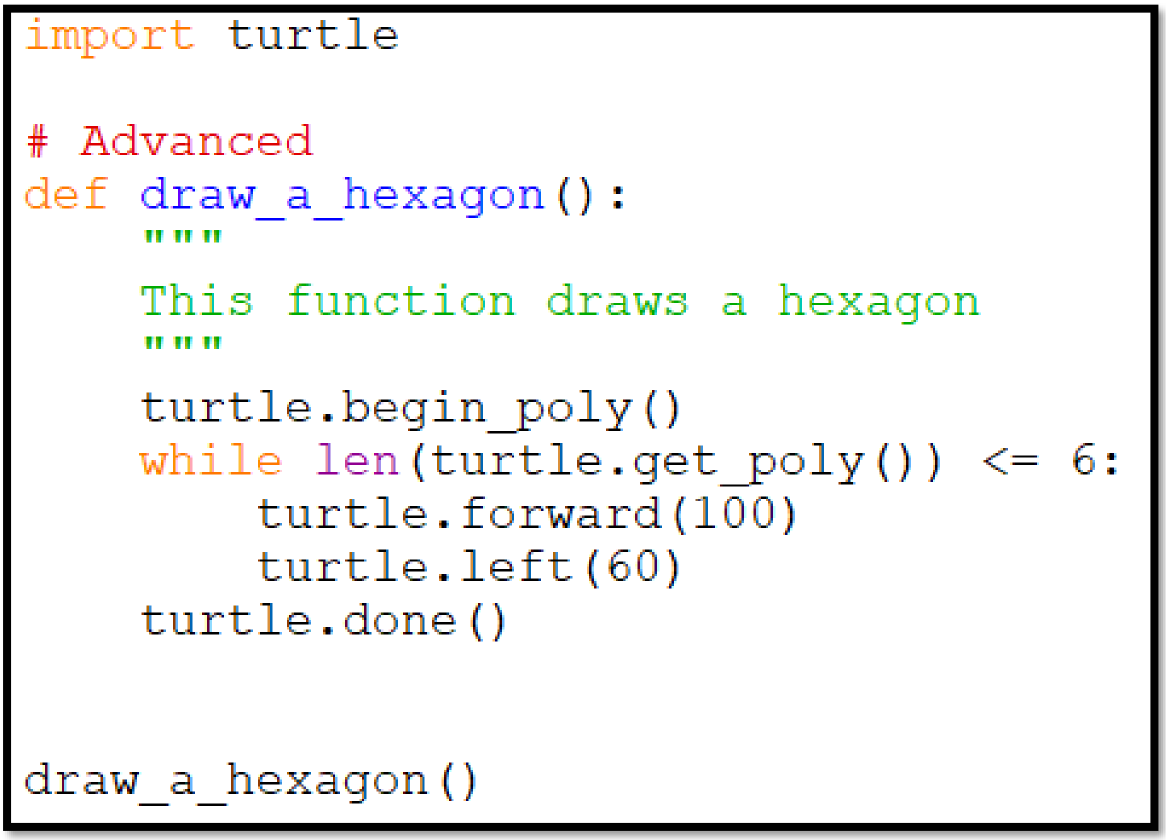
```
import turtle

# Beginner
turtle.forward(100)
turtle.left(60)
turtle.forward(100)
turtle.left(60)
turtle.forward(100)
turtle.left(60)
turtle.forward(100)
turtle.left(60)
turtle.forward(100)
turtle.left(60)
turtle.forward(100)
turtle.done()
```

Intermediate:



```python
import turtle

# Intermediate
number_of_sides = 6
sides_drawn = 0
while sides_drawn < number_of_sides:
    turtle.forward(100)
    turtle.left(60)
    sides_drawn += 1
turtle.done()
```

Advanced:



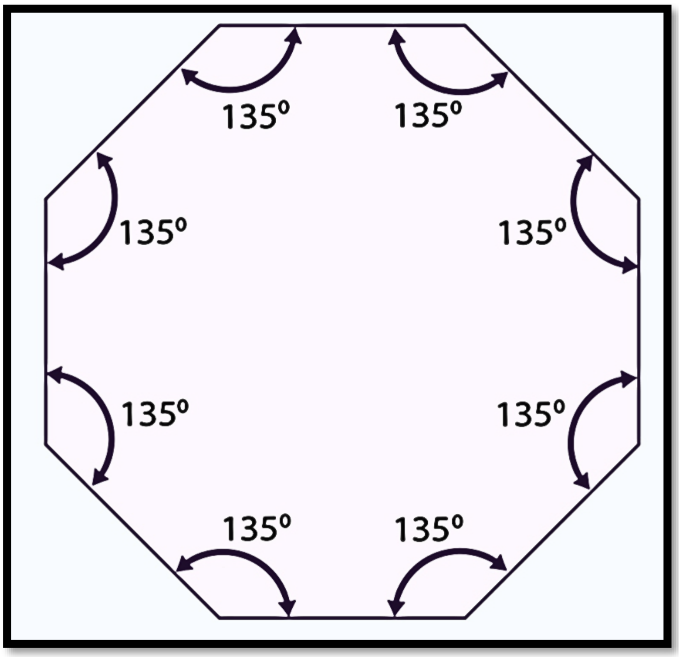
```python
import turtle

# Advanced
def draw_a_hexagon():
    """
    This function draws a hexagon
    """
    turtle.begin_poly()
    while len(turtle.get_poly()) <= 6:
        turtle.forward(100)
        turtle.left(60)
    turtle.done()


draw_a_hexagon()
```

How to Draw an Octagon

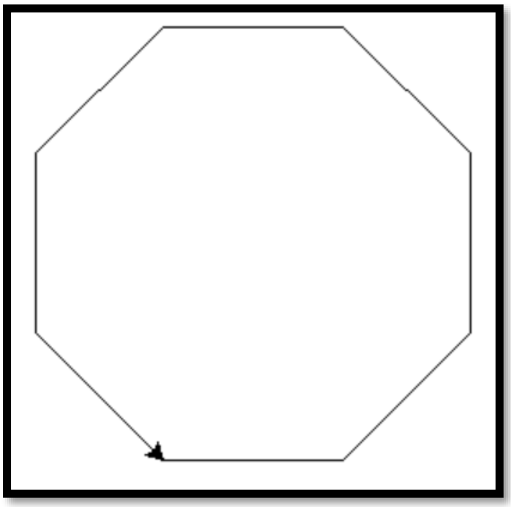
Step by Step

Starting from the bottom left corner:

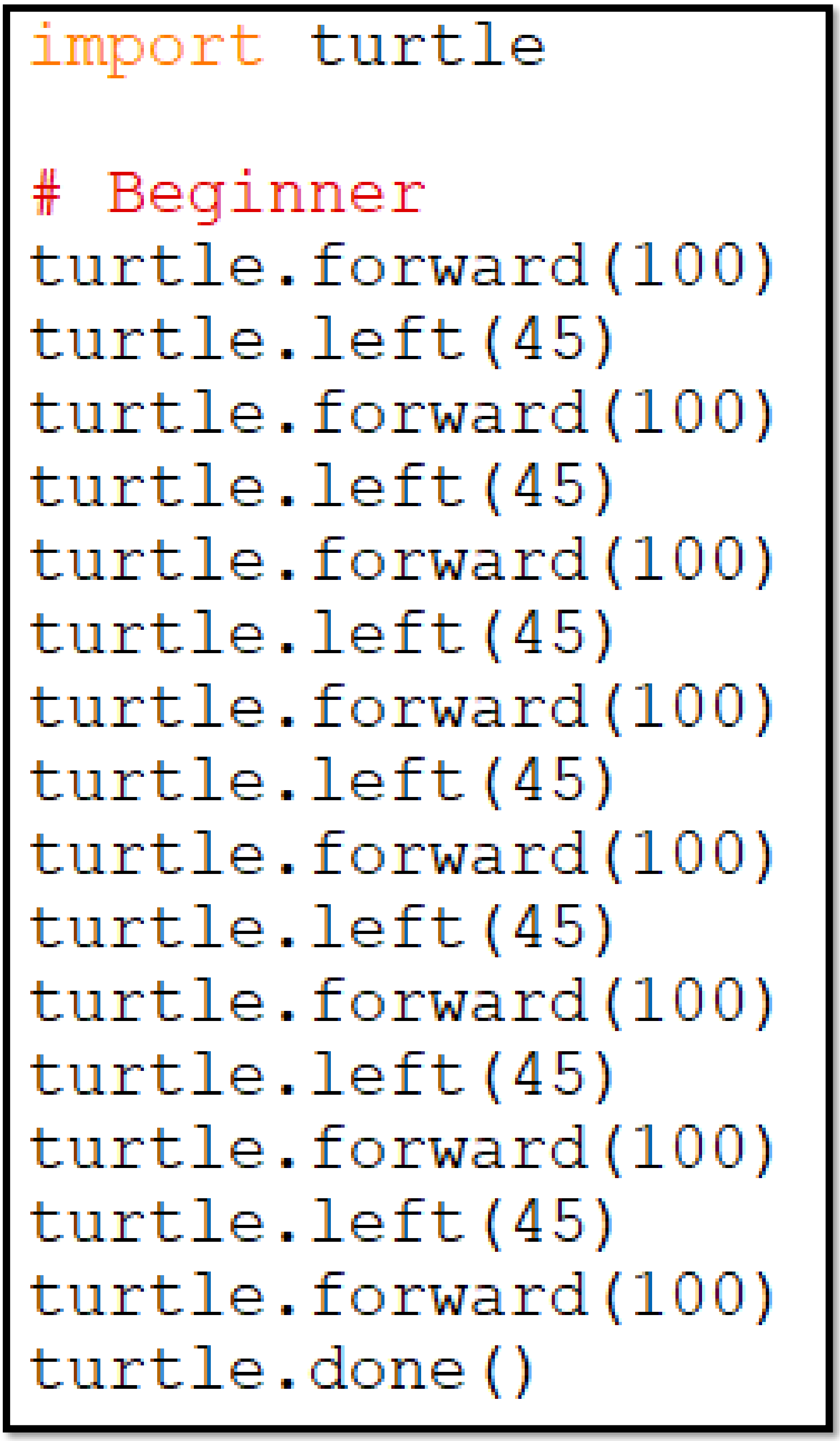
1. Draw a line of length 100.
2. Turn the pointer left by 45 degrees.
3. Repeat steps 1-2 7 more times.



Output:

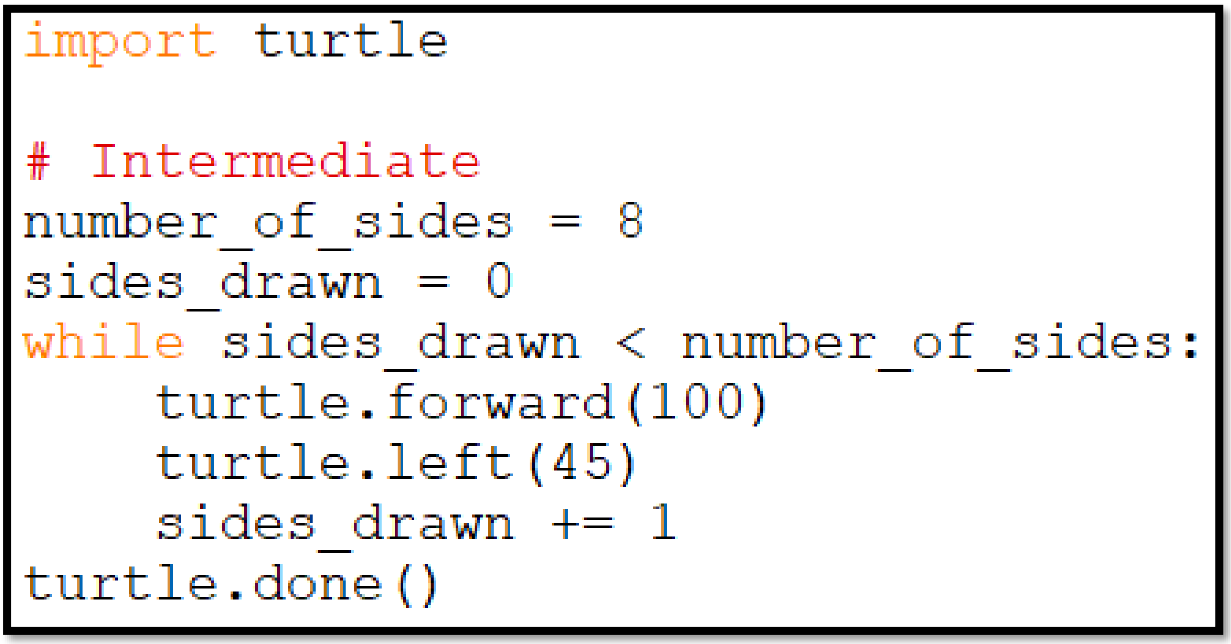


Beginner:



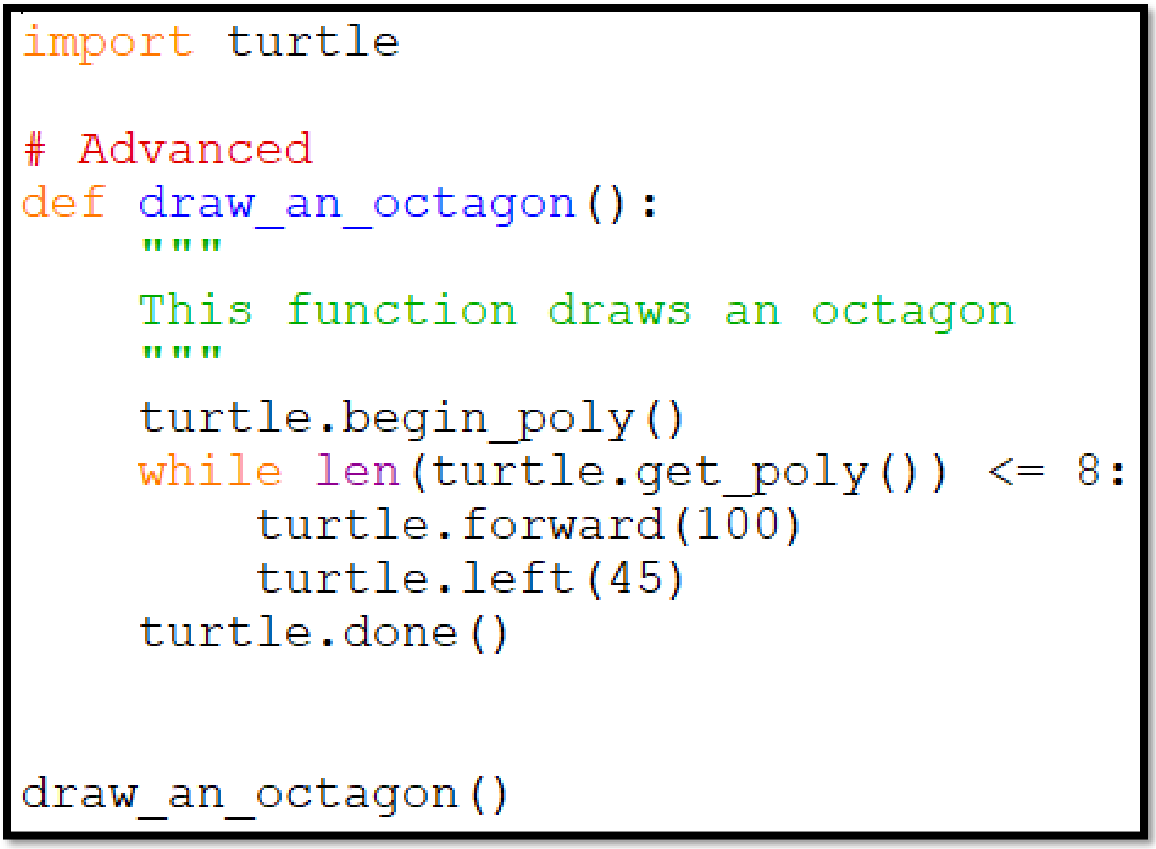
```
import turtle

# Beginner
turtle.forward(100)
turtle.left(45)
turtle.forward(100)
turtle.left(45)
turtle.forward(100)
turtle.left(45)
turtle.forward(100)
turtle.left(45)
turtle.forward(100)
turtle.left(45)
turtle.forward(100)
turtle.left(45)
turtle.forward(100)
turtle.left(45)
turtle.forward(100)
turtle.done()
```

Intermediate:



```
import turtle

# Intermediate
number_of_sides = 8
sides_drawn = 0
while sides_drawn < number_of_sides:
    turtle.forward(100)
    turtle.left(45)
    sides_drawn += 1
turtle.done()
```

Advanced:



```
import turtle

# Advanced
def draw_an_octagon():
    """
    This function draws an octagon
    """
    turtle.begin_poly()
    while len(turtle.get_poly()) <= 8:
        turtle.forward(100)
        turtle.left(45)
    turtle.done()


draw_an_octagon()
```

How to Draw a Rectangle

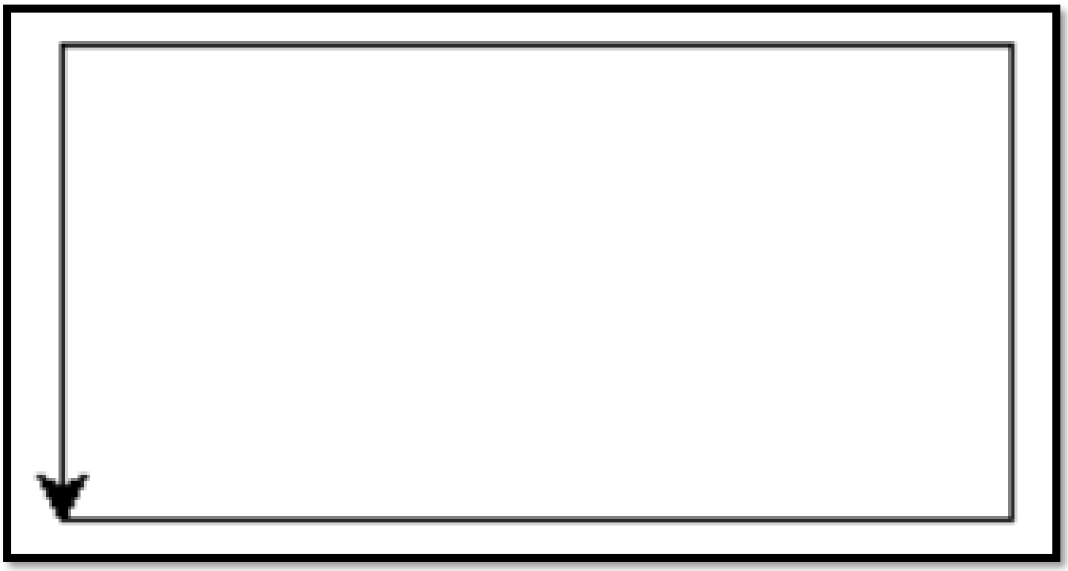
Step by Step:

Starting from the bottom left corner:

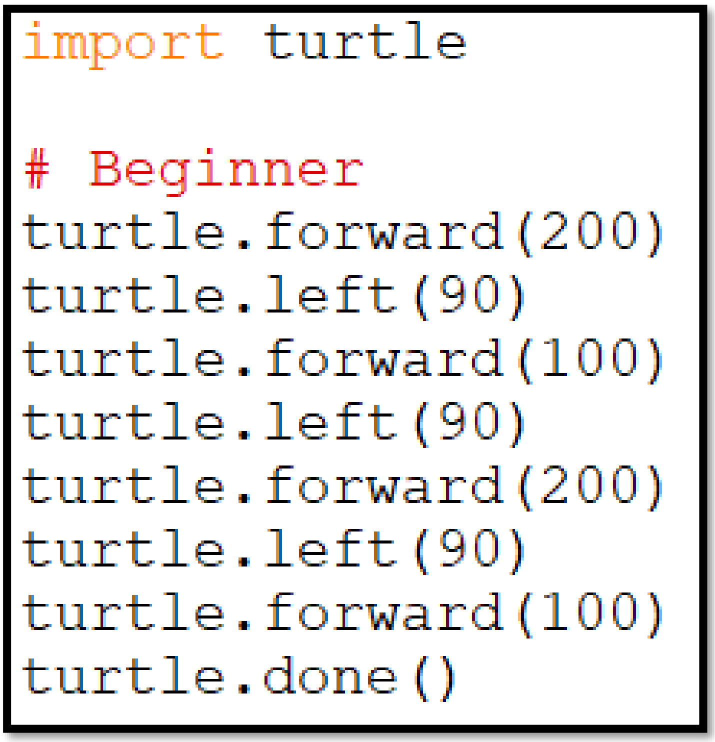
1. Draw a line of length 200.
2. Turn the pointer left by 90 degrees.
3. Draw a line of length 100.
4. Turn the pointer left by 90 degrees.
5. Repeat steps 1-4 one more time.

Rectangles are more complex to draw because not all their sides are the same length. In the intermediate and advanced example, a new concept is introduced. These are called conditional statements. They are another fundamental part of programming as they allow us to run different sections of code depending on the outcome of the statement. If you want to learn more about conditional statements, the following is a great resource https://www.learnpython.org/en/Conditions.

Output:

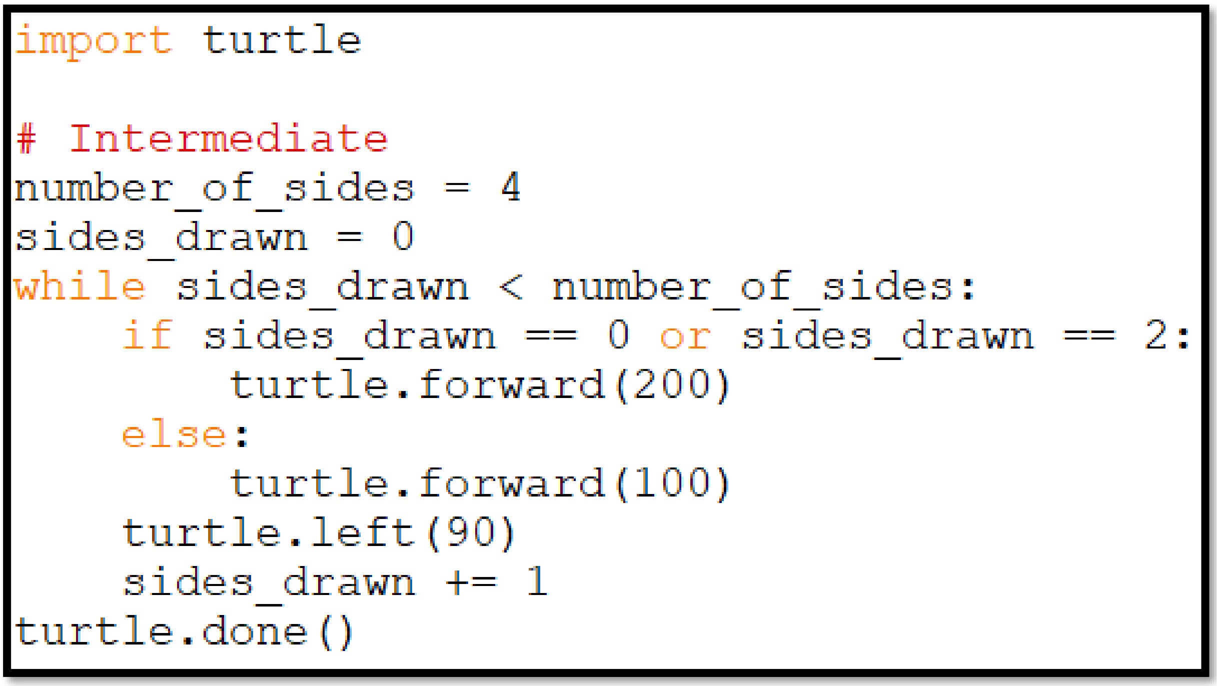


Beginner:



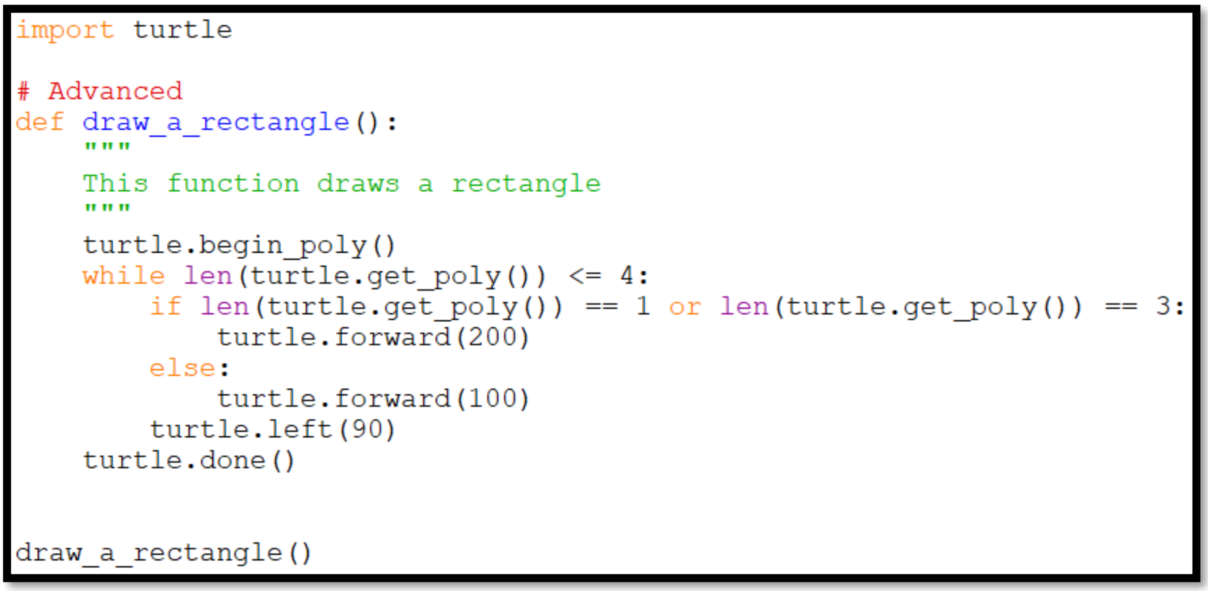
```
import turtle

# Beginner
turtle.forward(200)
turtle.left(90)
turtle.forward(100)
turtle.left(90)
turtle.forward(200)
turtle.left(90)
turtle.forward(100)
turtle.done()
```

Intermediate:



```
import turtle

# Intermediate
number_of_sides = 4
sides_drawn = 0
while sides_drawn < number_of_sides:
    if sides_drawn == 0 or sides_drawn == 2:
        turtle.forward(200)
    else:
        turtle.forward(100)
    turtle.left(90)
    sides_drawn += 1
turtle.done()
```

Advanced:



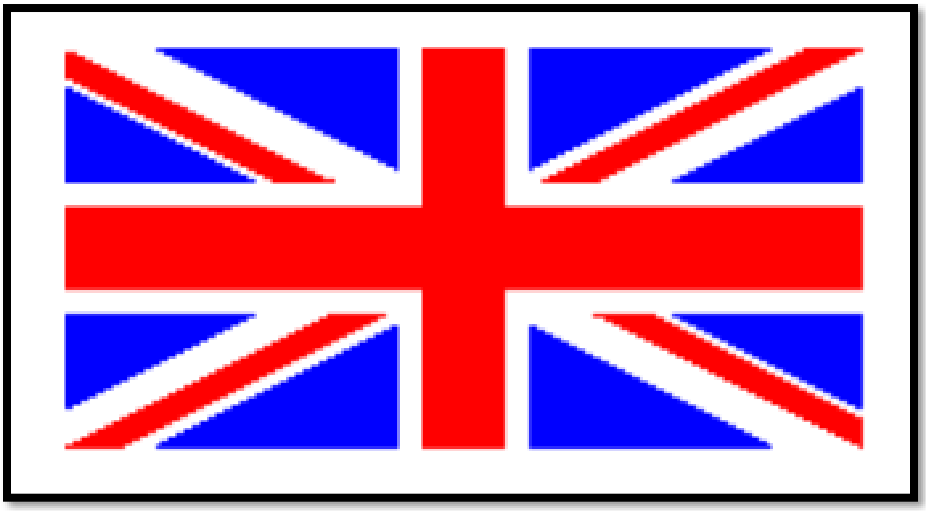
```python
import turtle

# Advanced
def draw_a_rectangle():
    """
    This function draws a rectangle
    """
    turtle.begin_poly()
    while len(turtle.get_poly()) <= 4:
        if len(turtle.get_poly()) == 1 or len(turtle.get_poly()) == 3:
            turtle.forward(200)
        else:
            turtle.forward(100)
        turtle.left(90)
    turtle.done()


draw_a_rectangle()
```

Challenge

Now it’s time for you to get creative.

You have all the tools now to draw something fantastic!

Here is an example of something that I drew using only the code included in this document:



Email me your creations to reubenj.mitchell@icloud.com.

(If you are under 16, ask your parents to email me).

About the Author

Hello, my name is Reuben Mitchell. I am currently studying Mechatronic and Robotic Engineering at the University of Sheffield. I have a passion for software development, and I am keen to share this with the next generation.

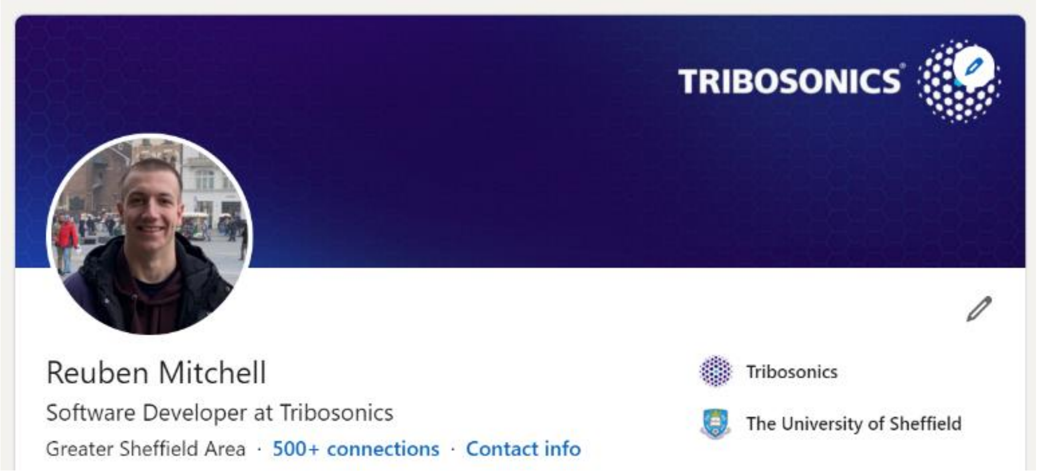
One of the first programming tools I learnt to use was Turtle, which is why I have created this tutorial with the aim to provide a gateway into programming.

Since then, I have used my software knowledge to make all kinds of cool projects: from a Sudoku solving app to an automated bar (pictured left) and along the way I have made Bluetooth controlled cars, image converter applications and even a robotic arm that I programmed to play checkers. But, most importantly, I've had a lot of fun! Software is a fantastic tool because the only limitation is your imagination. My hope is that after this tutorial, you will also share my passion!

Contact

Connect with me on LinkedIn: https://www.linkedin.com/in/reuben-mitchell-b9a7801a1/.

Or send me an email: reubenj.mitchell@icloud.com.



Reuben Mitchell

Software Developer at Tribosonics

Greater Sheffield Area · 500+ connections · Contact info