Python & Java {4} Teachers

ROBOTIIKKA- JA TIEDEKASVATUS RY

SCHOOL OF CODING

Elderberry

ROBOTIEM.LV

Robootika

BRADFIELD COLLEGE

Tessellations

Level 2 - Python

Funded by the
Erasmus+ Programme
of the European Union

{4}

Python & Java 4 Teachers

Funded by the
Erasmus+ Programme
of the European Union

Introduction

You probably think that art and programming couldn’t be more unalike, but actually they might be more similar than you think!

There are many modern artists whose main medium is code. For example: Mark Dorf, Josh Davis and Kyle McDonald.

Here are a few websites that merge together are and programming:

- Silk – Interactive Generative Art (weavesilk.com)
- Dream by WOMBO

Funded by the Erasmus+ Programme of the European Union

Task

- A tessellation is when a shape is repeated over and over again and fits together in a pattern without overlapping or leaving gaps.

- For this project we again will be needing the turtle graphics library again so that the shapes can be drawn.

Extension:

○ Complete the tessellation by drawing the other side of the shape.

Funded by the
Erasmus+ Programme
of the European Union

Process

This program should:

✓Import the python Turtle graphics Library

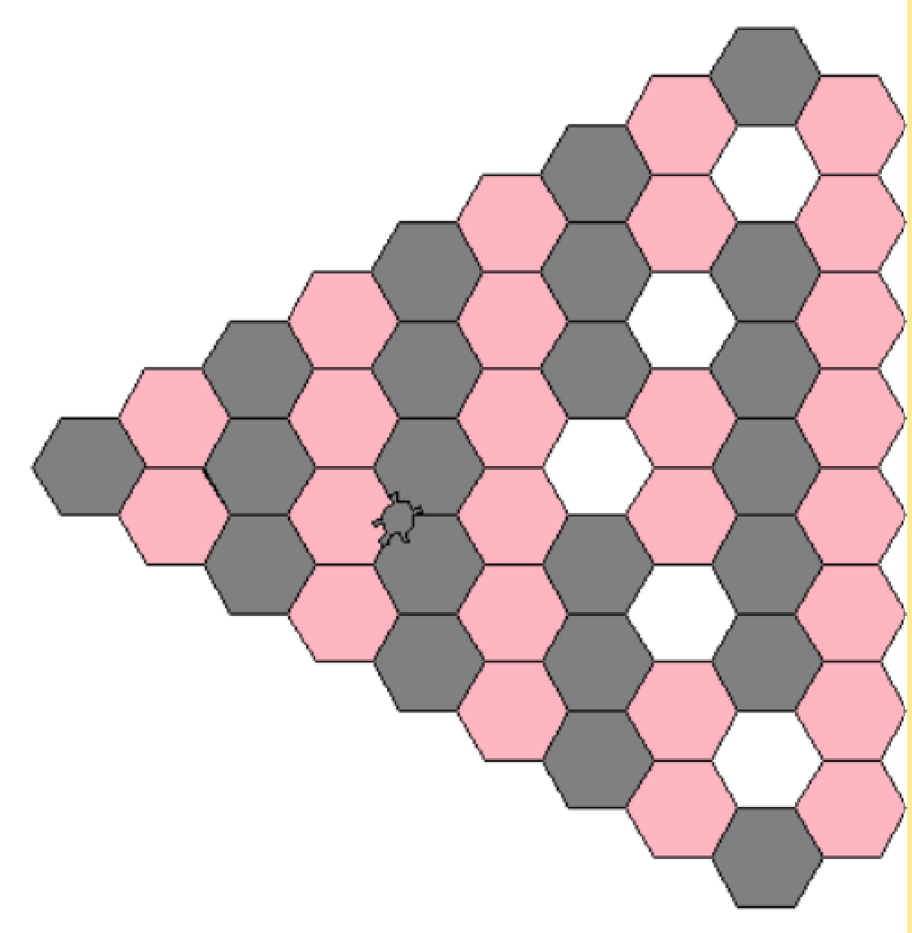
✓Control the speed fill colour and line color of the turtle,

✓Use for loops and subroutines to create the hexagonal shapes,

✓Apply previous knowledge of angles to help move the turtle along.

What it will look like…

Funded by the
Erasmus+ Programme
of the European Union



Funded by the Erasmus+ Programme of the European Union

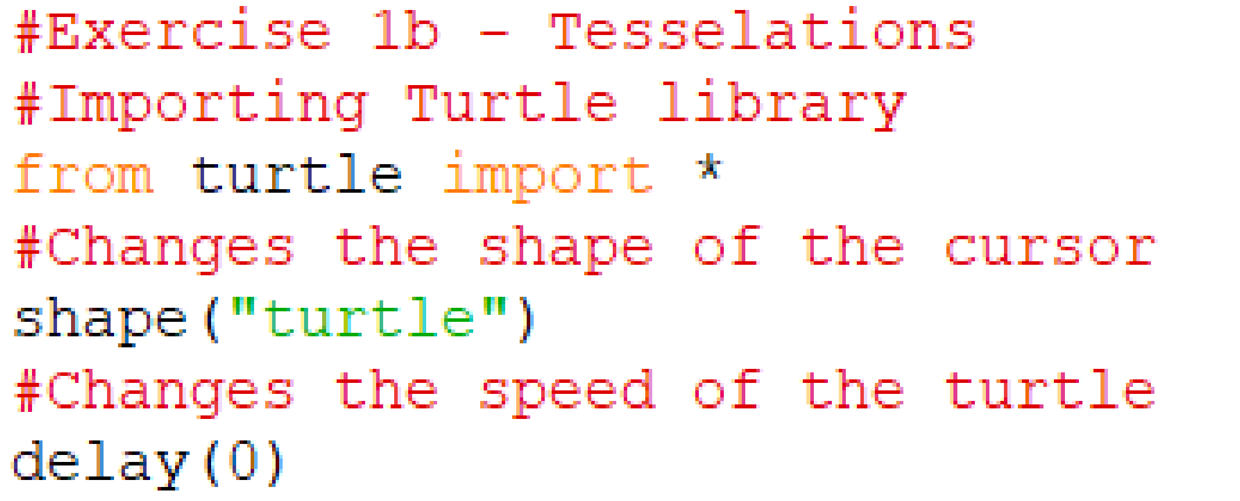
{4}

Python & Java 4 Teachers

Step 1

Import turtle Library

Here we have imported the Turtle library which mean we can now create graphics with the code we will write.

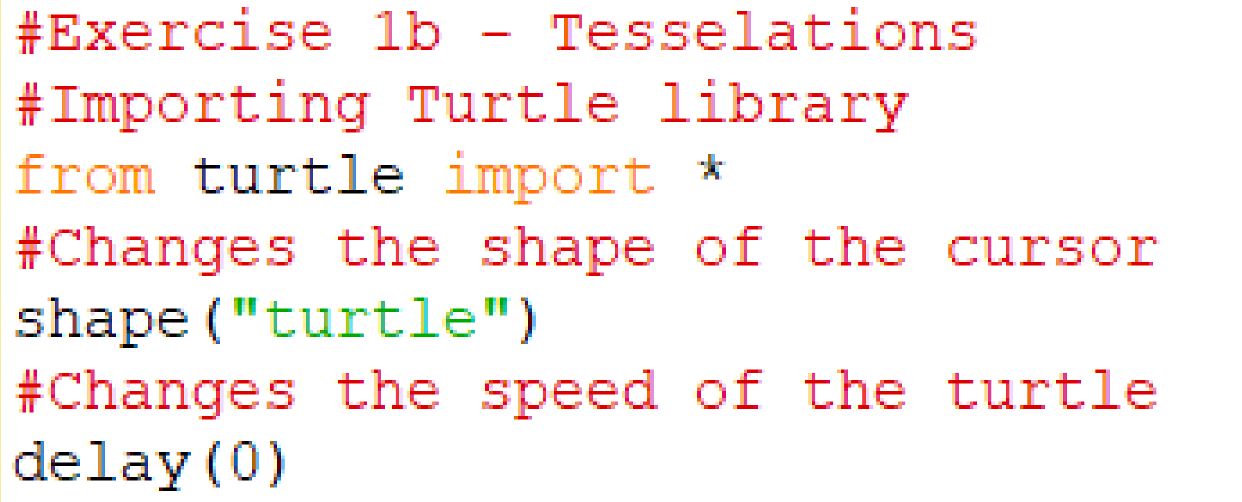


```
#Exercise 1b - Tesselations
#Importing Turtle library
from turtle import *
#Changes the shape of the cursor
shape("turtle")
#Changes the speed of the turtle
delay(0)
```

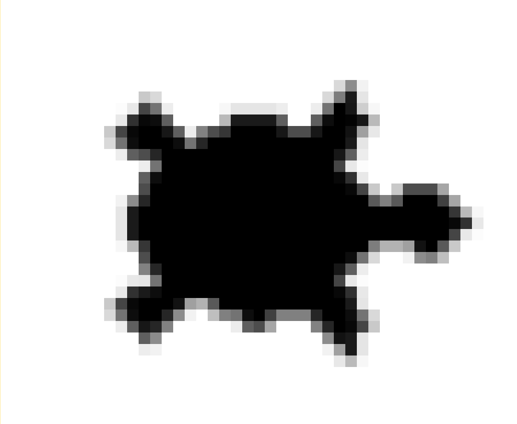
Here we have imported the Turtle graphics library, made the shape of the cursor a turtle and changed the speed of the turtle to go faster than its default speed.

Funded by the Erasmus+ Programme of the European Union

What it will look like…



```
#Exercise 1b - Tesselations
#Importing Turtle library
from turtle import *
#Changes the shape of the cursor
shape("turtle")
#Changes the speed of the turtle
delay(0)
```

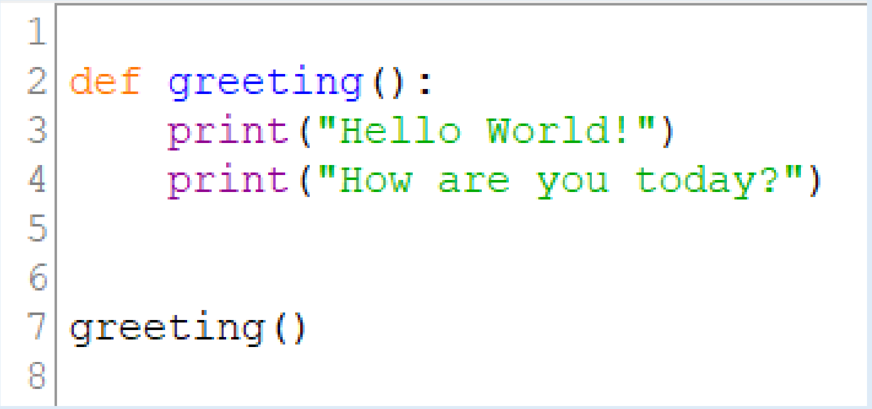


Subroutines

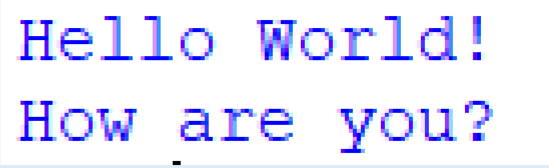
Funded by the
Erasmus+ Programme
of the European Union

{4}
Python & Java 4 Teachers

Subroutines are sets of instructions designed to perform a frequently used operation within a program.



```

def greeting():
    print("Hello World!")
    print("How are you today?")


greeting()

```



```
Hello World!
How are you?
```

Subroutines can store code and will only be run when ‘called’.

There are two main types of subroutine: procedures and functions.

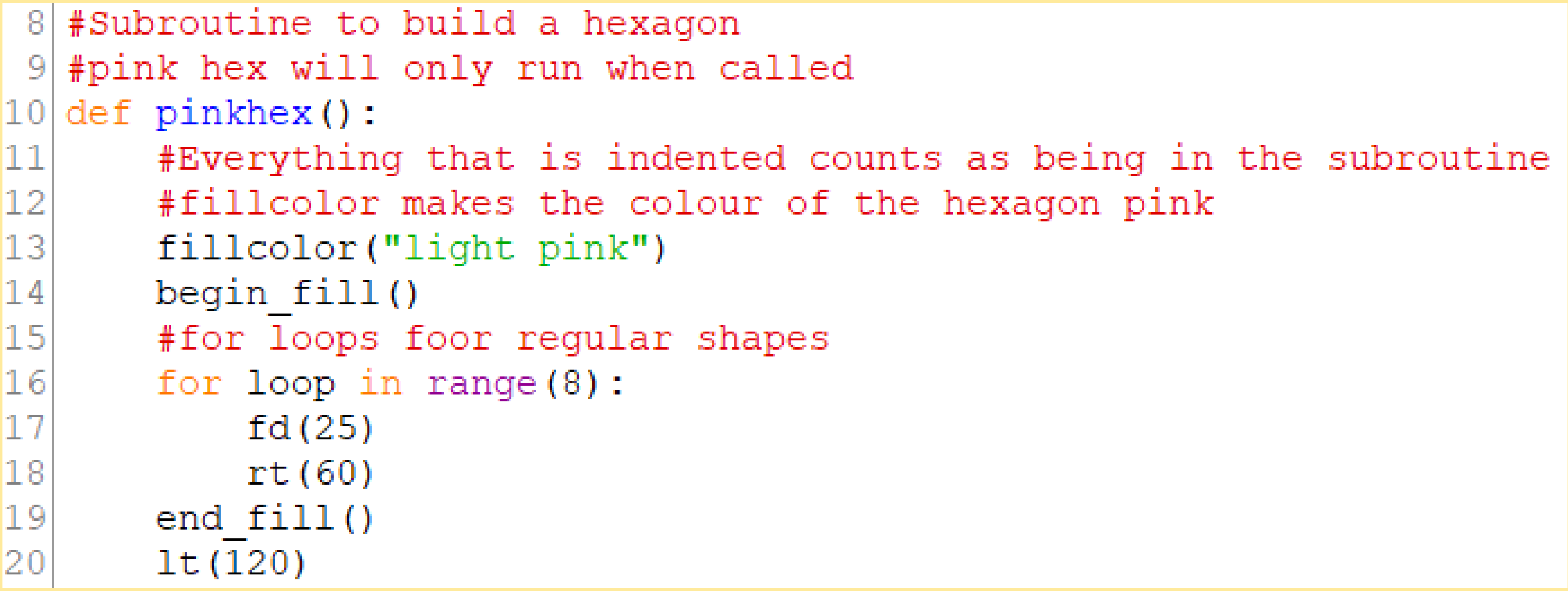
Procedures are not required to return a value, whereas functions must return a value.

Subroutines are great ways of writing more maintainable code and leads to more structured, organised and understandable programs.

Step 2

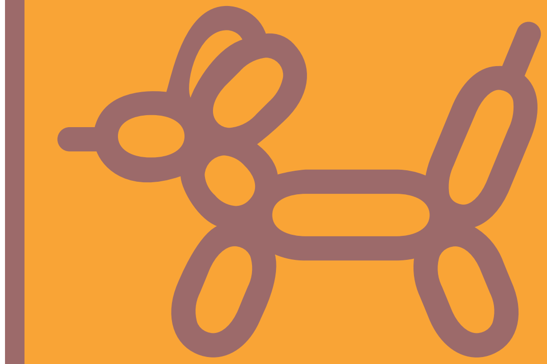
Creating a subroutine

In line 10 we create the first subroutine which includes the instructions for how to build a pink hexagon. To draw the shape itself a for loop is used to save writing out code unnecessarily



```
 8 #Subroutine to build a hexagon
 9 #pink hex will only run when called
10 def pinkhex():
11     #Everything that is indented counts as being in the subroutine
12     #fillcolor makes the colour of the hexagon pink
13     fillcolor("light pink")
14     begin_fill()
15     #for loops foor regular shapes
16     for loop in range(8):
17         fd(25)
18         rt(60)
19     end_fill()
20     lt(120)
```

Funded by the Erasmus+ Programme of the European Union



Loops

Funded by the
Erasmus+ Programme
of the European Union

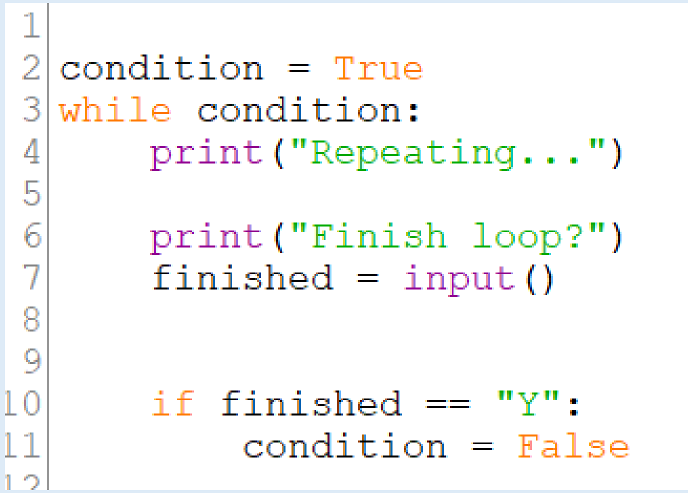
Python & Java 4 Teachers

A loop is a sequence of instructions that is continually repeated until a certain condition is reached.

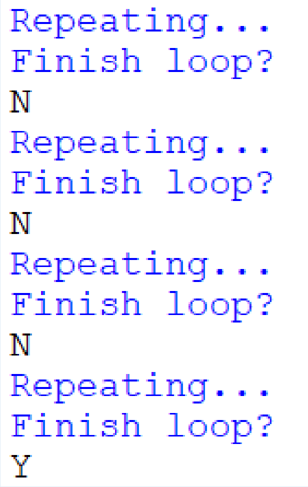
In Python there are two main loops: ‘FOR Loops’ and ‘WHILE Loops’

While Loops are condition controlled and will repeat until their condition is false.

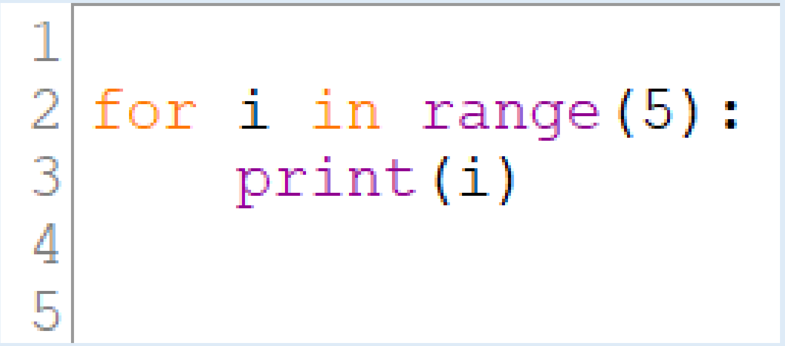
For loops are count controlled and will repeat a set number of times.



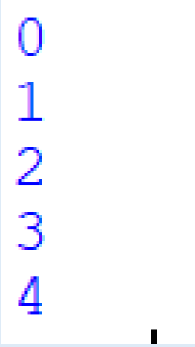
```python
condition = True
while condition:
    print("Repeating...")

    print("Finish loop?")
    finished = input()


    if finished == "Y":
        condition = False
```



```
Repeating...
Finish loop?
N
Repeating...
Finish loop?
N
Repeating...
Finish loop?
N
Repeating...
Finish loop?
Y
```



```python
for i in range(5):
    print(i)
```

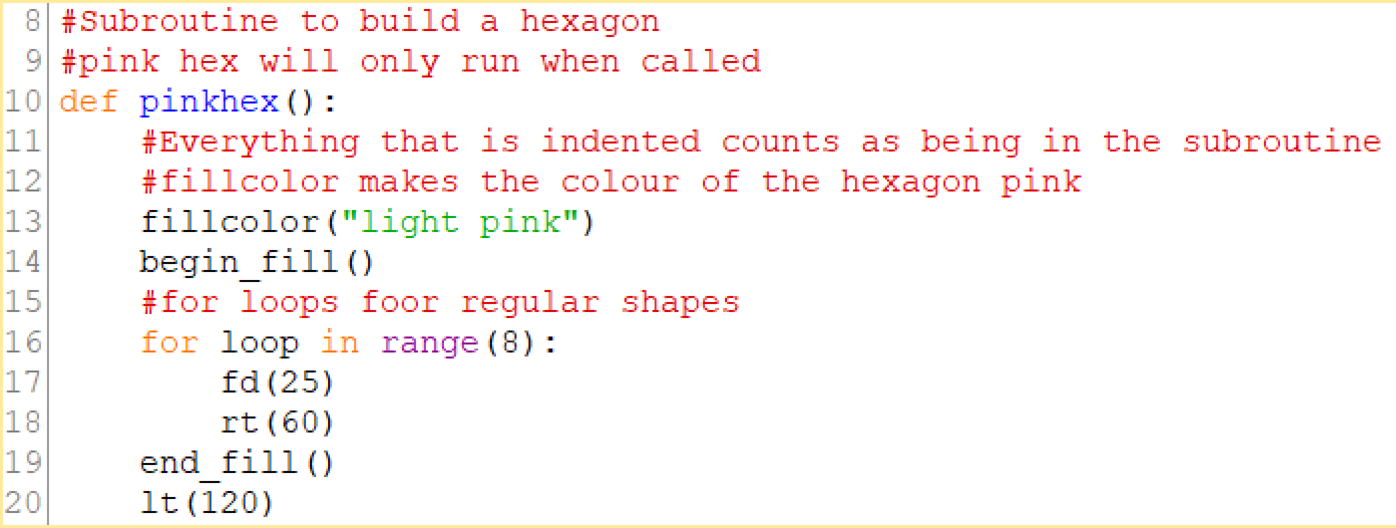


```
0
1
2
3
4
```

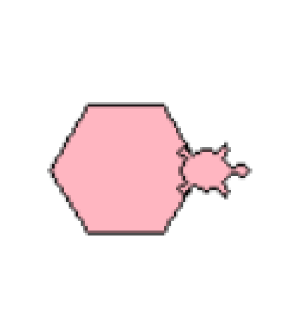
Funded by the
Erasmus+ Programme
of the European Union

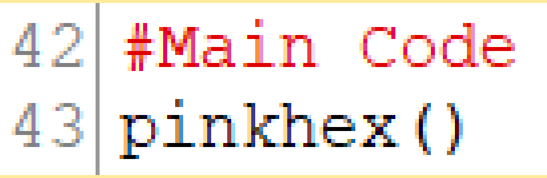
{4}
Python & Java 4 Teachers

What it will look like…



```
#Subroutine to build a hexagon
#pink hex will only run when called
def pinkhex():
    #Everything that is indented counts as being in the subroutine
    #fillcolor makes the colour of the hexagon pink
    fillcolor("light pink")
    begin_fill()
    #for loops foor regular shapes
    for loop in range(8):
        fd(25)
        rt(60)
    end_fill()
    lt(120)
```



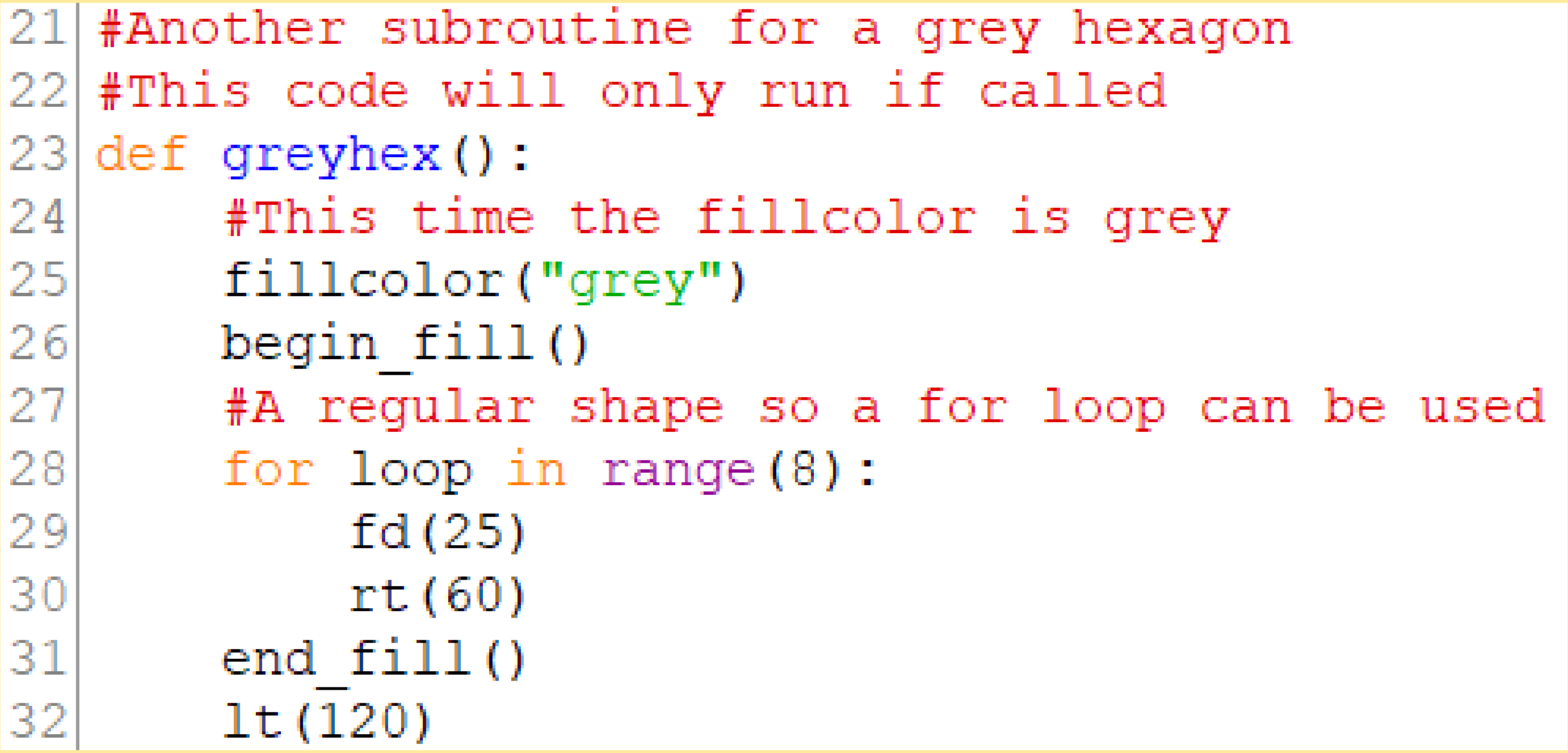


```
#Main Code
pinkhex()
```

As the program above is part of a subroutine so it needs to be called in the main code.

Step 3

Creating a second subroutine.



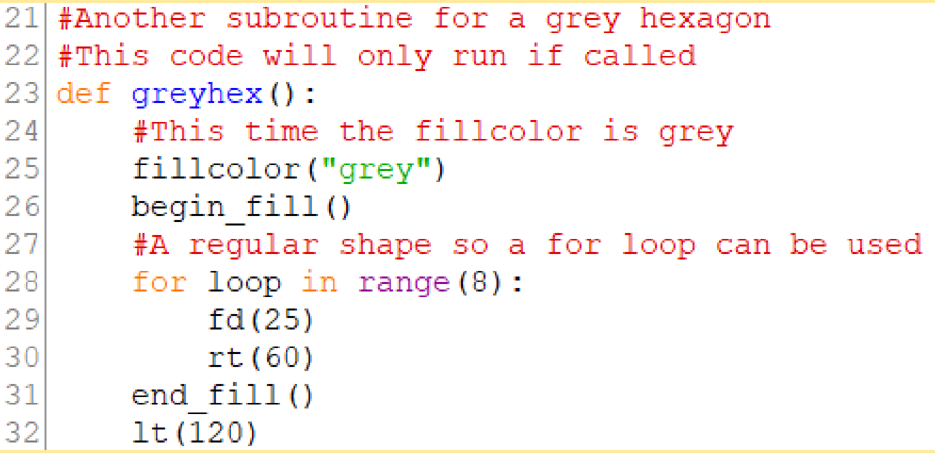
```
#Another subroutine for a grey hexagon
#This code will only run if called
def greyhex():
    #This time the fillcolor is grey
    fillcolor("grey")
    begin_fill()
    #A regular shape so a for loop can be used
    for loop in range(8):
        fd(25)
        rt(60)
    end_fill()
    lt(120)
```

This code creates another subroutine which creates a grey hexagon. The left turn at the end is needed so that when we call the subroutine in the main code the shapes won’t overlap.

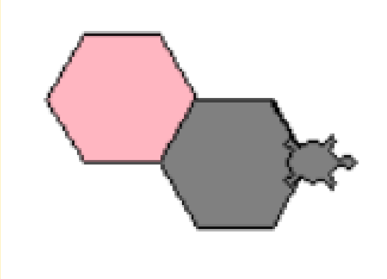
Funded by the
Erasmus+ Programme
of the European Union

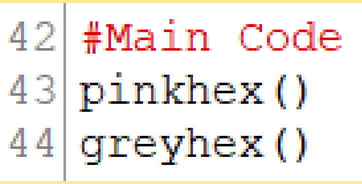
Funded by the
Erasmus+ Programme
of the European Union

What it will look like…



```python
#Another subroutine for a grey hexagon
#This code will only run if called
def greyhex():
    #This time the fillcolor is grey
    fillcolor("grey")
    begin_fill()
    #A regular shape so a for loop can be used
    for loop in range(8):
        fd(25)
        rt(60)
    end_fill()
    lt(120)
```



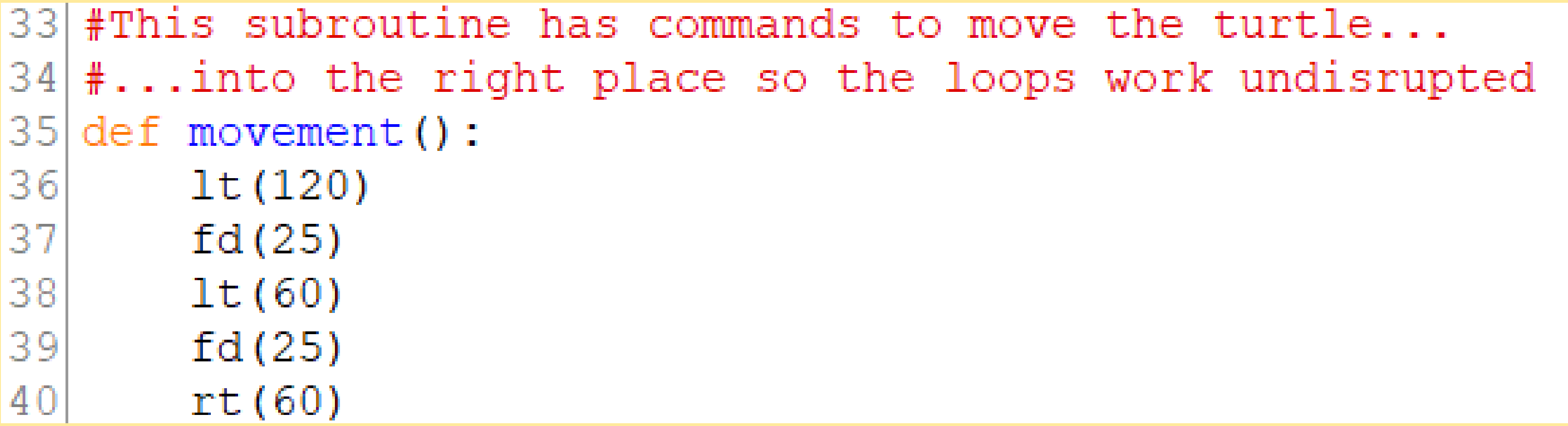


```python
#Main Code
pinkhex()
greyhex()
```

Funded by the
Erasmus+ Programme
of the European Union

Step 4

Creating the last subroutine

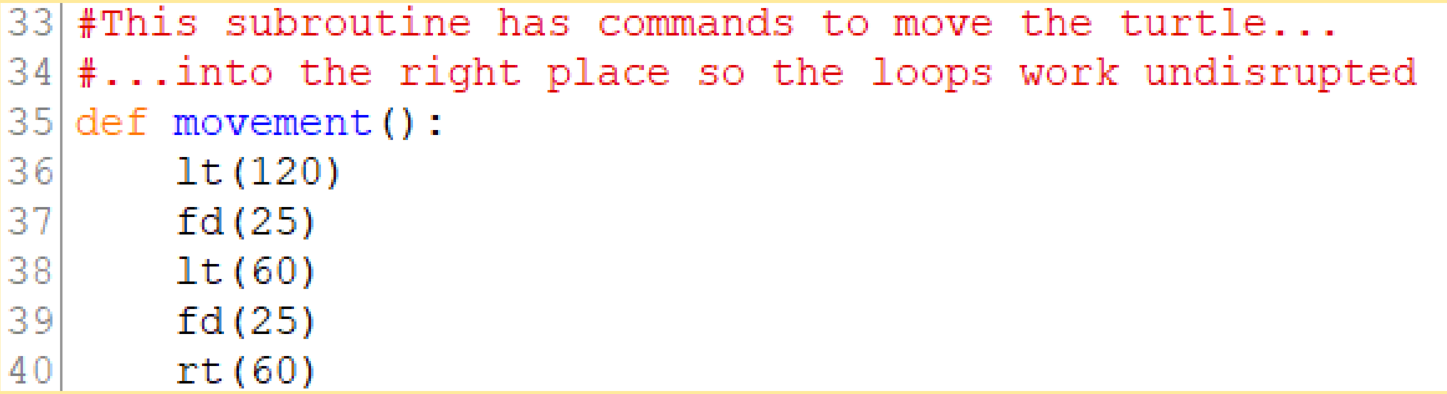


```
#This subroutine has commands to move the turtle...
#...into the right place so the loops work undisrupted
def movement():
    lt(120)
    fd(25)
    lt(60)
    fd(25)
    rt(60)
```

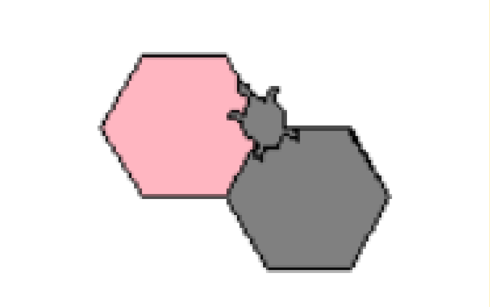
This is the last subroutine needed, but instead of drawing a shape it moves the turtle so that when the next instruction is called it is in the right place.

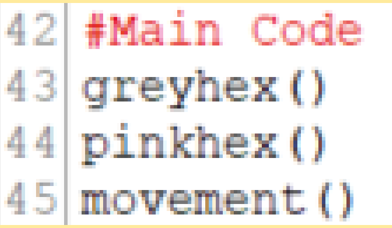
Funded by the Erasmus+ Programme of the European Union

What it will look like…



```python
#This subroutine has commands to move the turtle...
#...into the right place so the loops work undisrupted
def movement():
    lt(120)
    fd(25)
    lt(60)
    fd(25)
    rt(60)
```





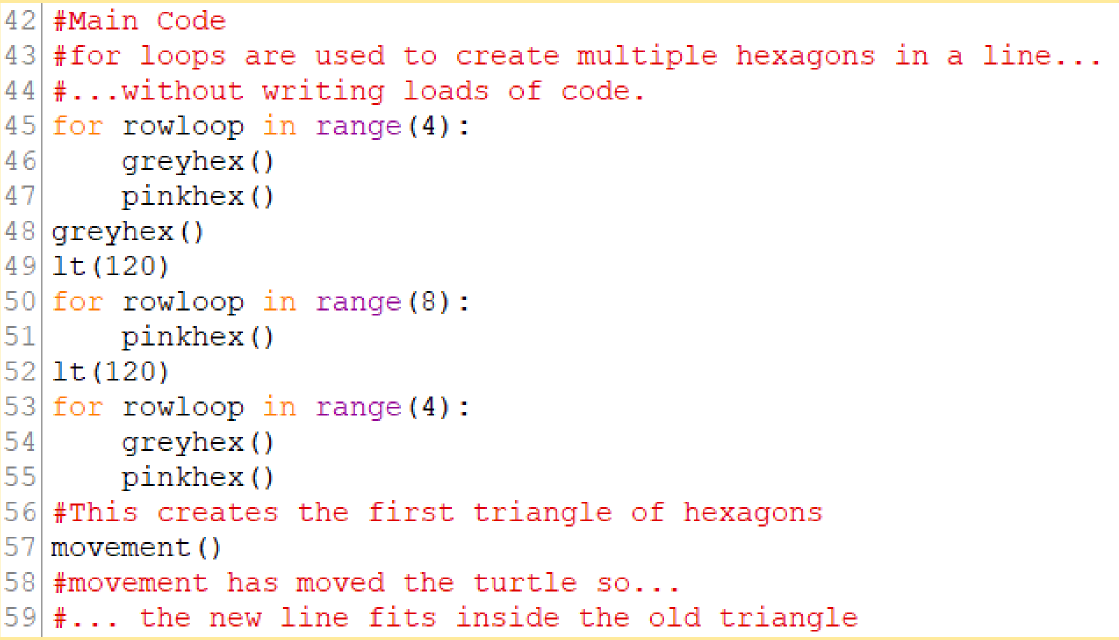
```python
#Main Code
greyhex()
pinkhex()
movement()
```

This section of the code was only used to show what the subroutines do when called.
IT ISN’T PART OF THE COMPLETE CODE!!!

Funded by the
Erasmus+ Programme
of the European Union

Step 5

Start of the main code



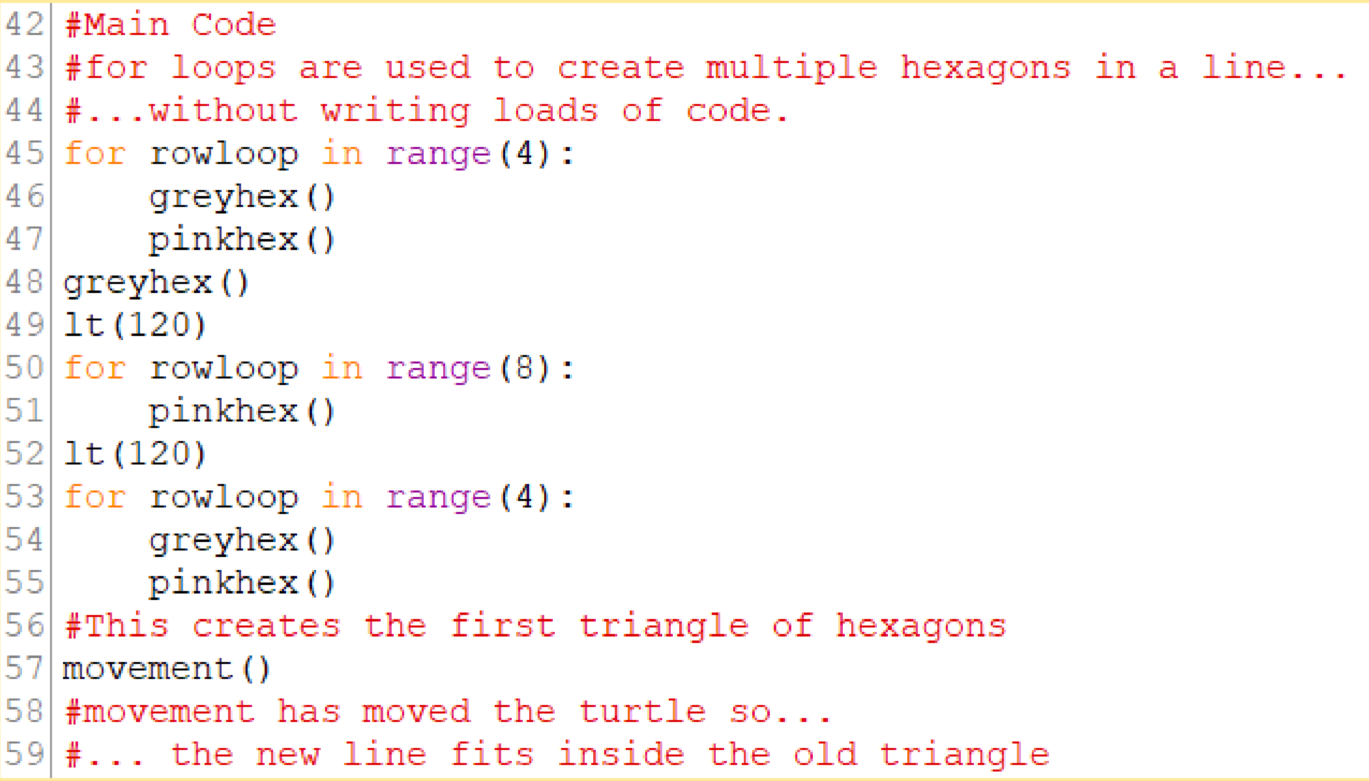
```
#Main Code
#for loops are used to create multiple hexagons in a line...
#...without writing loads of code.
for rowloop in range(4):
    greyhex()
    pinkhex()
greyhex()
lt(120)
for rowloop in range(8):
    pinkhex()
lt(120)
for rowloop in range(4):
    greyhex()
    pinkhex()
#This creates the first triangle of hexagons
movement()
#movement has moved the turtle so...
#... the new line fits inside the old triangle
```

This is the body of the main code. For loops are used to create the triangular shape of hexagons.

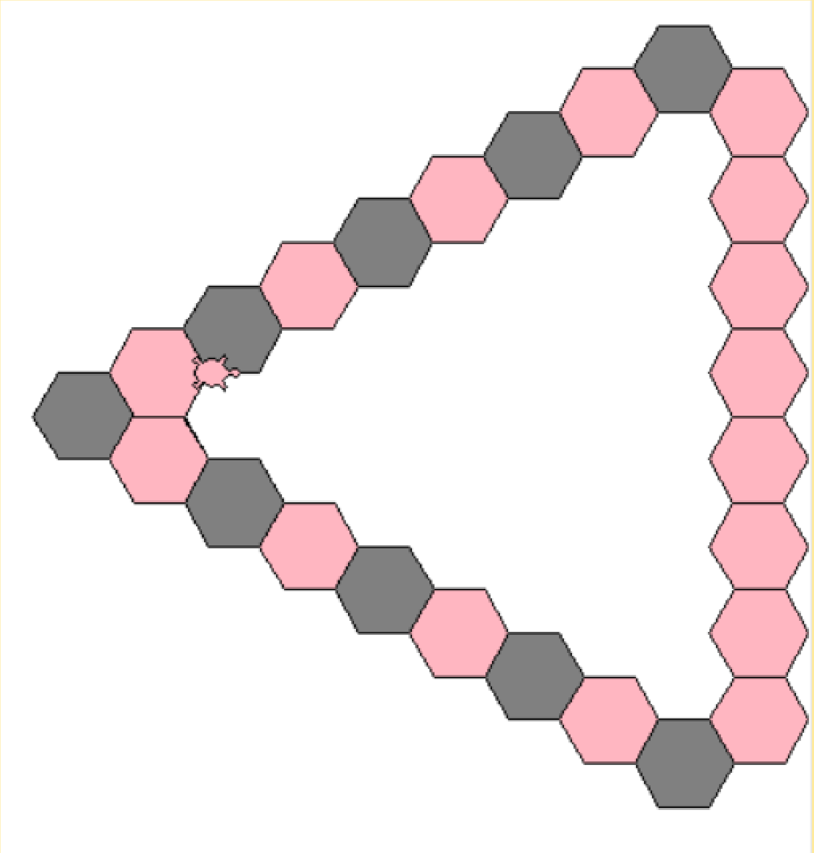
Funded by the Erasmus+ Programme of the European Union

Python & Java 4 Teachers

What it will look like…



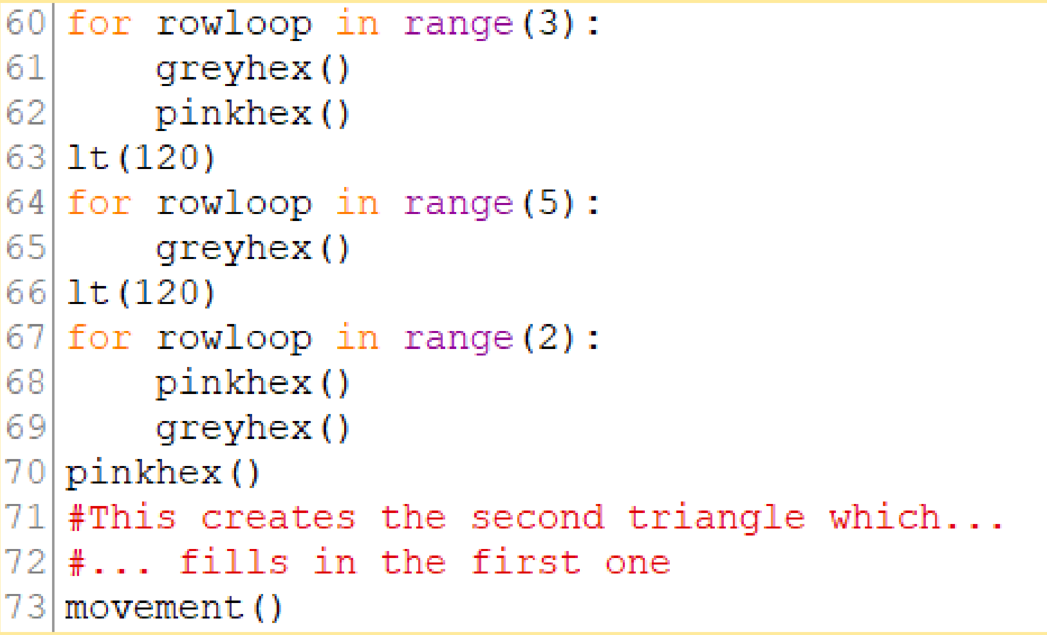
```python
#Main Code
#for loops are used to create multiple hexagons in a line...
#...without writing loads of code.
for rowloop in range(4):
    greyhex()
    pinkhex()
greyhex()
lt(120)
for rowloop in range(8):
    pinkhex()
lt(120)
for rowloop in range(4):
    greyhex()
    pinkhex()
#This creates the first triangle of hexagons
movement()
#movement has moved the turtle so...
#... the new line fits inside the old triangle
```



Funded by the
Erasmus+ Programme
of the European Union

Step 6

The main code pt2



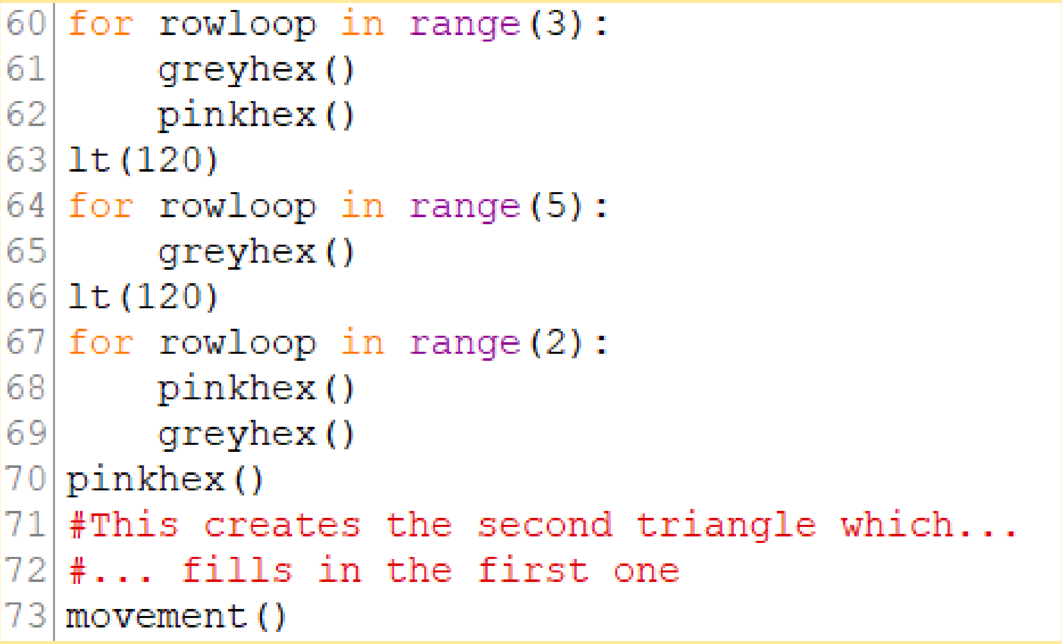
```
for rowloop in range(3):
    greyhex()
    pinkhex()
lt(120)
for rowloop in range(5):
    greyhex()
lt(120)
for rowloop in range(2):
    pinkhex()
    greyhex()
pinkhex()
#This creates the second triangle which...
#... fills in the first one
movement()
```

We repeat the previous steps but decrease the numbers within each for loop so it fits inside the first triangle.

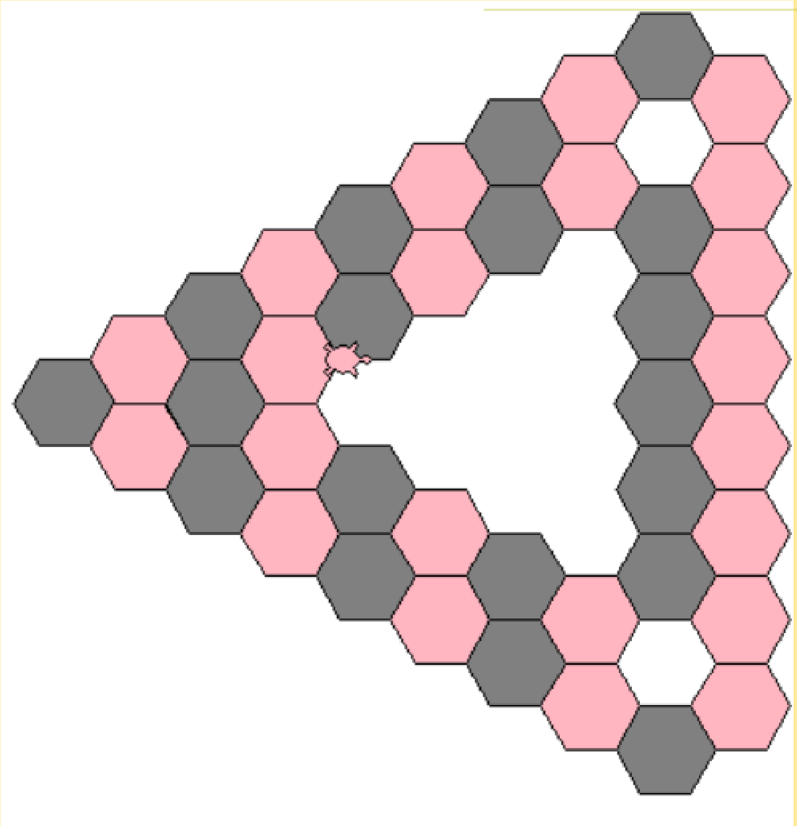
Funded by the
Erasmus+ Programme
of the European Union

Python & Java 4 Teachers

What will it look like…

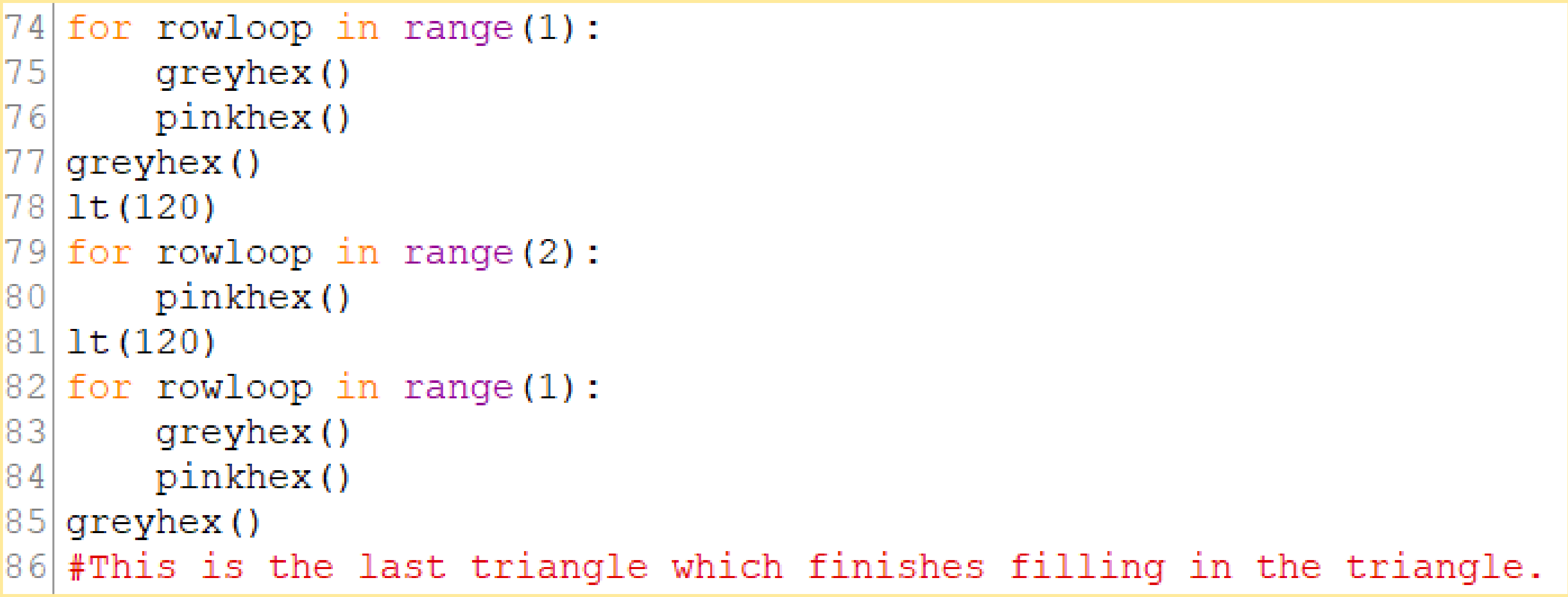


```
for rowloop in range(3):
    greyhex()
    pinkhex()
lt(120)
for rowloop in range(5):
    greyhex()
lt(120)
for rowloop in range(2):
    pinkhex()
    greyhex()
pinkhex()
#This creates the second triangle which...
#... fills in the first one
movement()
```



Step 7

The main code pt3



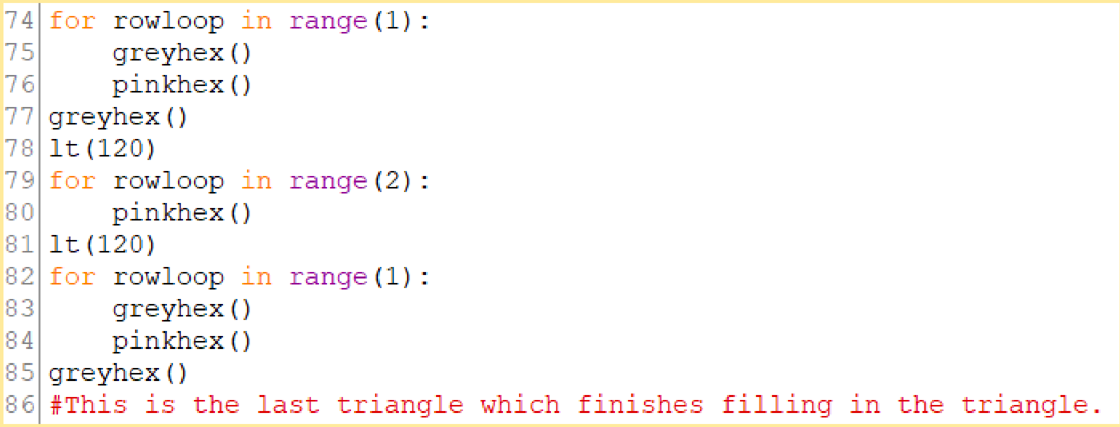
```
for rowloop in range(1):
    greyhex()
    pinkhex()
greyhex()
lt(120)
for rowloop in range(2):
    pinkhex()
lt(120)
for rowloop in range(1):
    greyhex()
    pinkhex()
greyhex()
#This is the last triangle which finishes filling in the triangle.
```

This is the last set of for loops that fills in the rest of the triangle.

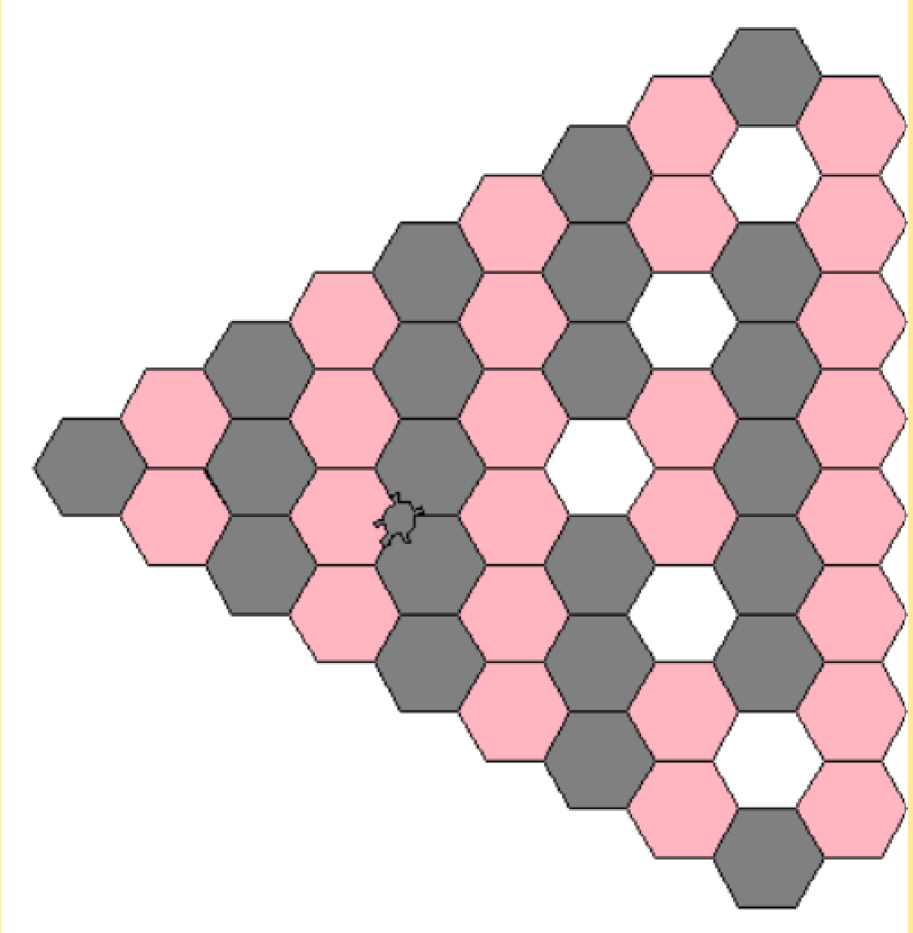
Funded by the
Erasmus+ Programme
of the European Union

Funded by the Erasmus+ Programme of the European Union

What it will look like…



```python
for rowloop in range(1):
    greyhex()
    pinkhex()
greyhex()
lt(120)
for rowloop in range(2):
    pinkhex()
lt(120)
for rowloop in range(1):
    greyhex()
    pinkhex()
greyhex()
#This is the last triangle which finishes filling in the triangle.
```

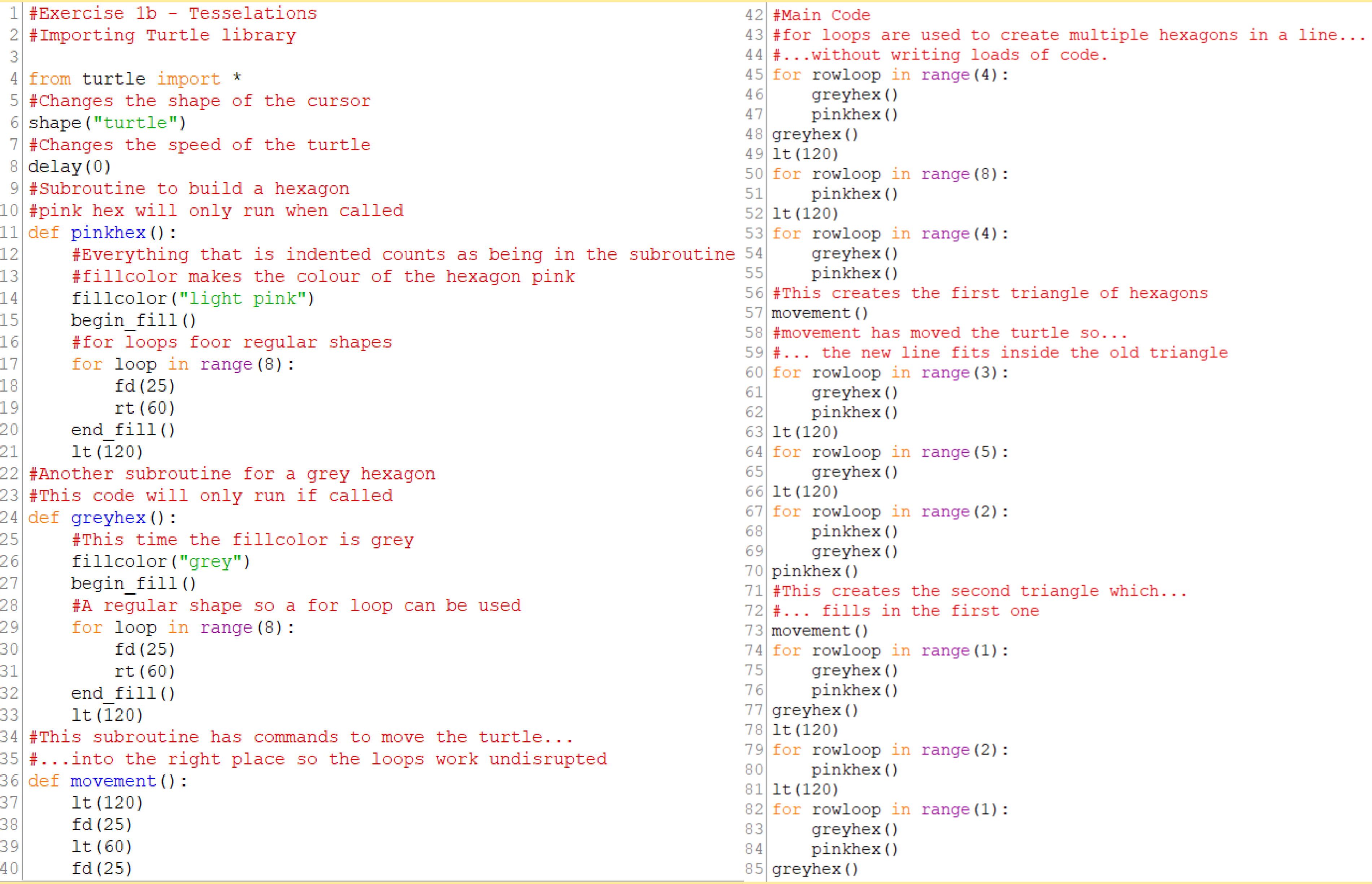


CHALLENGE:

- Finish the tessellation by drawing the reflection of the shape – so it forms a bowtie shape.
- Or you can create your own tessellation with different shapes and patterns.

Final Product:

Funded by the
Erasmus+ Programme
of the European Union



```
#Exercise 1b - Tesselations
#Importing Turtle library

from turtle import *
#Changes the shape of the cursor
shape("turtle")
#Changes the speed of the turtle
delay(0)
#Subroutine to build a hexagon
#pink hex will only run when called
def pinkhex():
    #Everything that is indented counts as being in the subroutine
    #fillcolor makes the colour of the hexagon pink
    fillcolor("light pink")
    begin_fill()
    #for loops foor regular shapes
    for loop in range(8):
        fd(25)
        rt(60)
    end_fill()
    lt(120)
#Another subroutine for a grey hexagon
#This code will only run if called
def greyhex():
    #This time the fillcolor is grey
    fillcolor("grey")
    begin_fill()
    #A regular shape so a for loop can be used
    for loop in range(8):
        fd(25)
        rt(60)
    end_fill()
    lt(120)
#This subroutine has commands to move the turtle...
#...into the right place so the loops work undisrupted
def movement():
    lt(120)
    fd(25)
    lt(60)
    fd(25)
```

```
#Main Code
#for loops are used to create multiple hexagons in a line...
#...without writing loads of code.
for rowloop in range(4):
    greyhex()
    pinkhex()
greyhex()
lt(120)
for rowloop in range(8):
    pinkhex()
lt(120)
for rowloop in range(4):
    greyhex()
    pinkhex()
#This creates the first triangle of hexagons
movement()
#movement has moved the turtle so...
#... the new line fits inside the old triangle
for rowloop in range(3):
    greyhex()
    pinkhex()
lt(120)
for rowloop in range(5):
    greyhex()
lt(120)
for rowloop in range(2):
    pinkhex()
    greyhex()
pinkhex()
#This creates the second triangle which...
#... fills in the first one
movement()
for rowloop in range(1):
    greyhex()
    pinkhex()
greyhex()
lt(120)
for rowloop in range(2):
    pinkhex()
lt(120)
for rowloop in range(1):
    greyhex()
    pinkhex()
greyhex()
```

Funded by the
Erasmus+ Programme
of the European Union

{4}

Python & Java 4 Teachers

Conclusion

This program should:

- ✓You should have confidently been able to import a library into Python,
- ✓You should be confident in using subroutines to create regular shapes.
- ✓You should be comfortable using angles to navigate through regular shapes,
- ✓You should be confident in manipulating the speed and colour within a turtle graphics program.

Congratulations!

You have created a tessellation program

Funded by the
Erasmus+ Programme
of the European Union