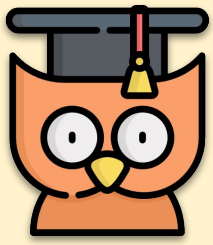
[the academy_of_code]

Grade 2
Unit 4

Python Programming

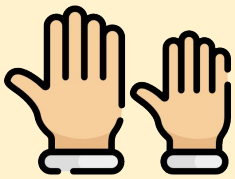
www.theacademyofcode.com/handouts

Lesson 1 - Input/Output Devices

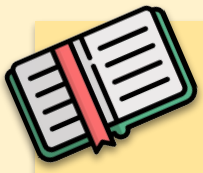


Learning Outcomes:

- Treating computers with care
- Input and Output devices
- Design a poster
- Coding Basics - Lightbot



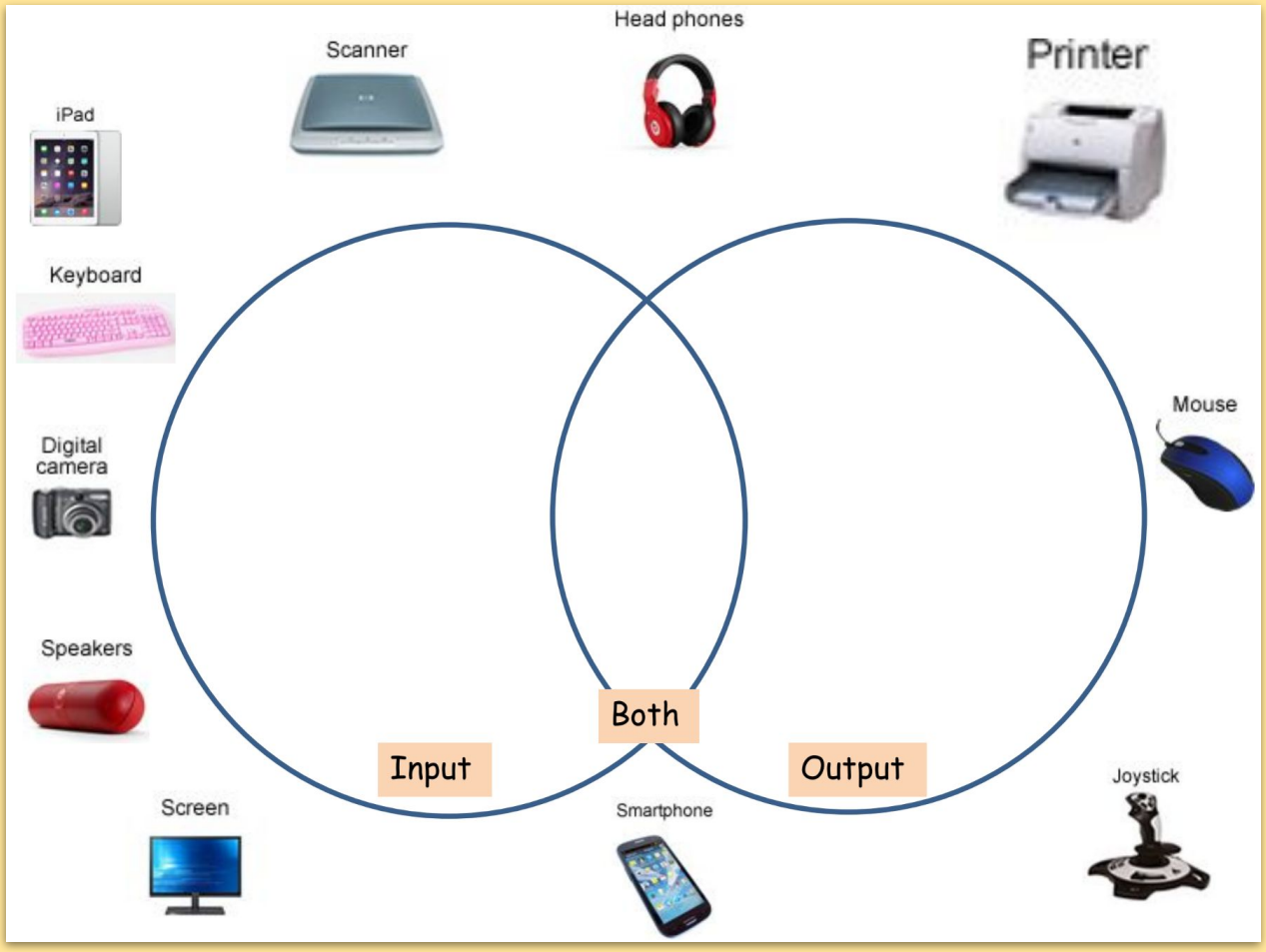
REMEMBER: Put up your hand. We love to help!



Input/ Output Devices

Some devices input data into the computer. Some devices output data.

Have a look at the tech items below. In which categories do they belong?



Talking Time

Steven says “a keyboard is an input device.” Is he Correct? Why?

Carol says “a printer is an input device.” Is she Correct? Why?

A teacher uses a projector to show you their presentations :

Is it an input device or output device?

Why?

Playing on a games console (PlayStation/Xbox):

What is an Input device used to play on it?

What is an output device you could use?

How many input and output devices can you see in this room?

What are they?

How many have we used so far today?

Game Time - Typing Skills

We've had an awesome 6 weeks with HTML and CSS. We're about to start into a module on Python which will involve a bit of typing. Lets polish up on our typing skills. Today's three typing activities are:

- **BBC Dancemat**
  - https://www.taoc.ie/bbcdm1
  - https://www.taoc.ie/bbcdm2
  - https://www.taoc.ie/bbcdm3
- **Keyboard Climber**
  - https://www.taoc.ie/keyboardclimber
- **Typing Attack**
  - https://taoc.ie/typeattack

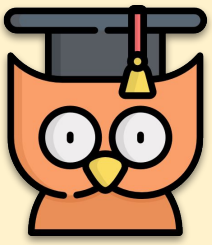
Game Time - An Introduction to Python

For the remainder of the class, we're going to use **python** to battle through the dungeon in **CodeCombat.** This is a fun way to practice and learn the python programming language to a high high level.

You can play at this link

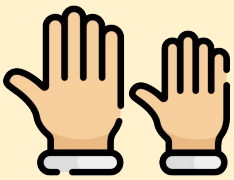
www.codecombat.com/play/dungeon

Lesson 2 - Block Turtle I

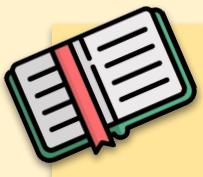


Learning Outcomes:

- Meeting and talking about Tina
- Making Tina draw some simple shapes in Block Turtle
- Working out different locations for Tina to get to on a map.
- Drawing multiple circles.



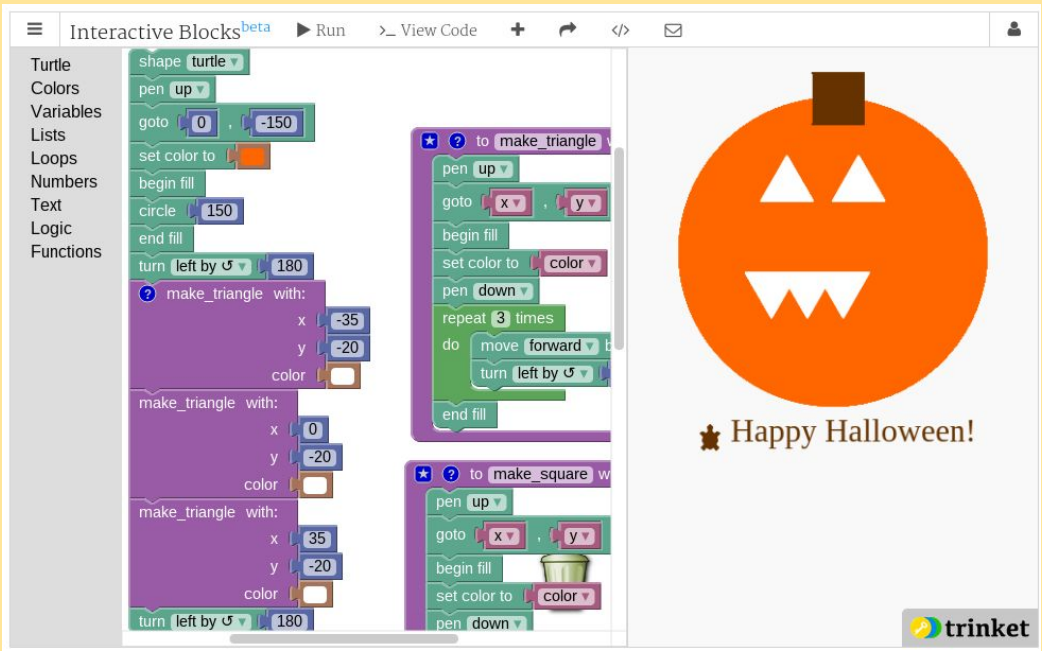
REMEMBER: Put up your hand. We love to help!



Block Turtle

Today we're going to get started with coding with **Python Blocks**. We can create images and graphics in **Python Blocks** with **Tina the Turtle**.

Tina the turtle moves around your screen just like the mouse arrow, but unlike the mouse, Tina leaves a trail behind her. Tina can draw all kinds of shapes and pictures - you just need to give them the right commands.

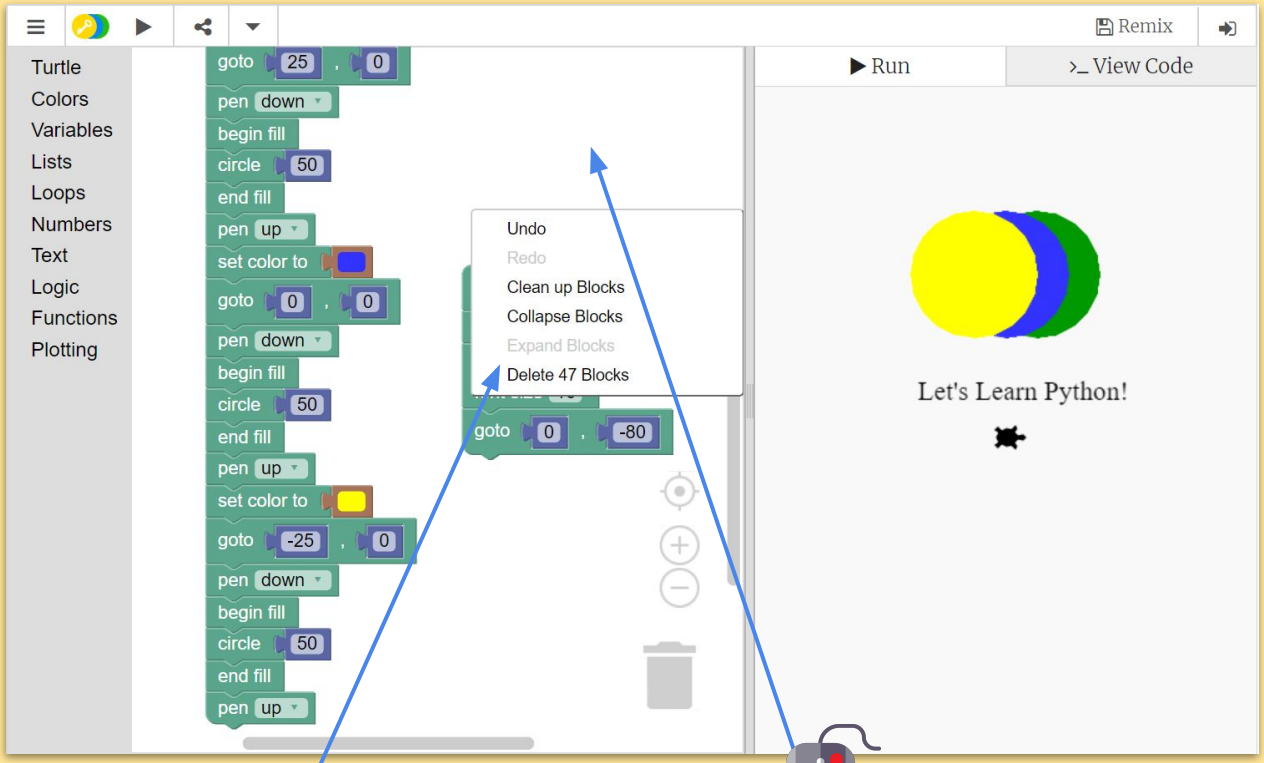


A very advanced drawing example. Can you see Tina?

Let's Get Coding

1 Open Trinket, if it isn't already open. **CTRL** + **CLICK**

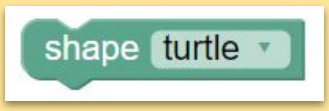
www.bit.ly/blankblocktrinket



2 If there are blocks already here, Right Click on this white area and Left Click on 'Delete Blocks' to clean up your **workspace**.

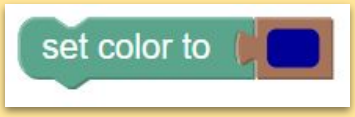
Now we're ready to code. We are going to draw a simple shape (a circle) using Tina the Turtle.

3 Set the style of the turtle with a function called shape.



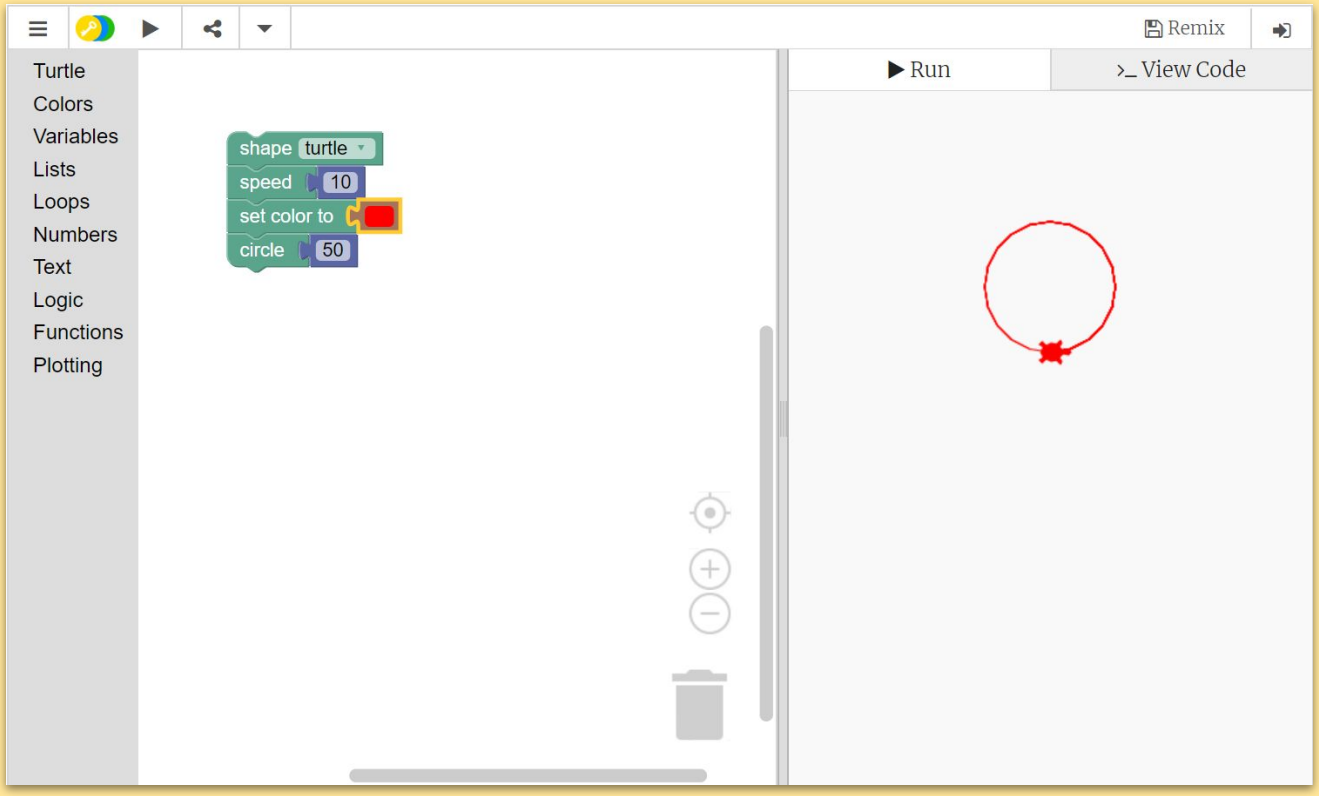
4 Set the turtle's speed, choosing a number between 1 and 10 (10 is the fastest).

5 Set the colour that the turtle will use to draw the shape.

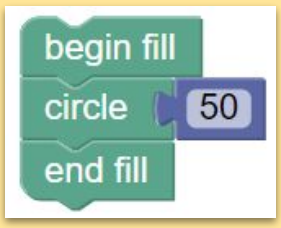


6 Now, tell your turtle to draw a circle.

7 Run the code above. You should get a result like this:



8 Fill the inside of the circle with a colour (same colour as the outline) using the following block.



Challenge

If you've finished this tasks, play around with the other blocks for a few minutes to see what else you can do with Tina.

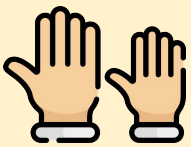
See if you can:

- Make Tina draw **another circle**
  - in a different colour
  - In a different position
- Make Tina **write some text**
- Try and position a few circles in different areas (we're doing this next week, so don't worry if you can't).

Lesson 3 - Block Turtle II

Learning Outcomes:

- Working out different locations for Tina to get to on a map.
- Drawing multiple circles and different x & y coordinates



REMEMBER: Put up your hand. We love to help!

Game Time

For the first few minutes, have a go at whatever games you didn't try in last week's warm up.

- **BBC Dancemat**
  - https://www.taoc.ie/bbcdm1
  - https://www.taoc.ie/bbcdm2
  - https://www.taoc.ie/bbcdm3
- **Keyboard Climber**
  - https://www.taoc.ie/keyboardclimber
- **Typing Attack**
  - https://taoc.ie/typeattack

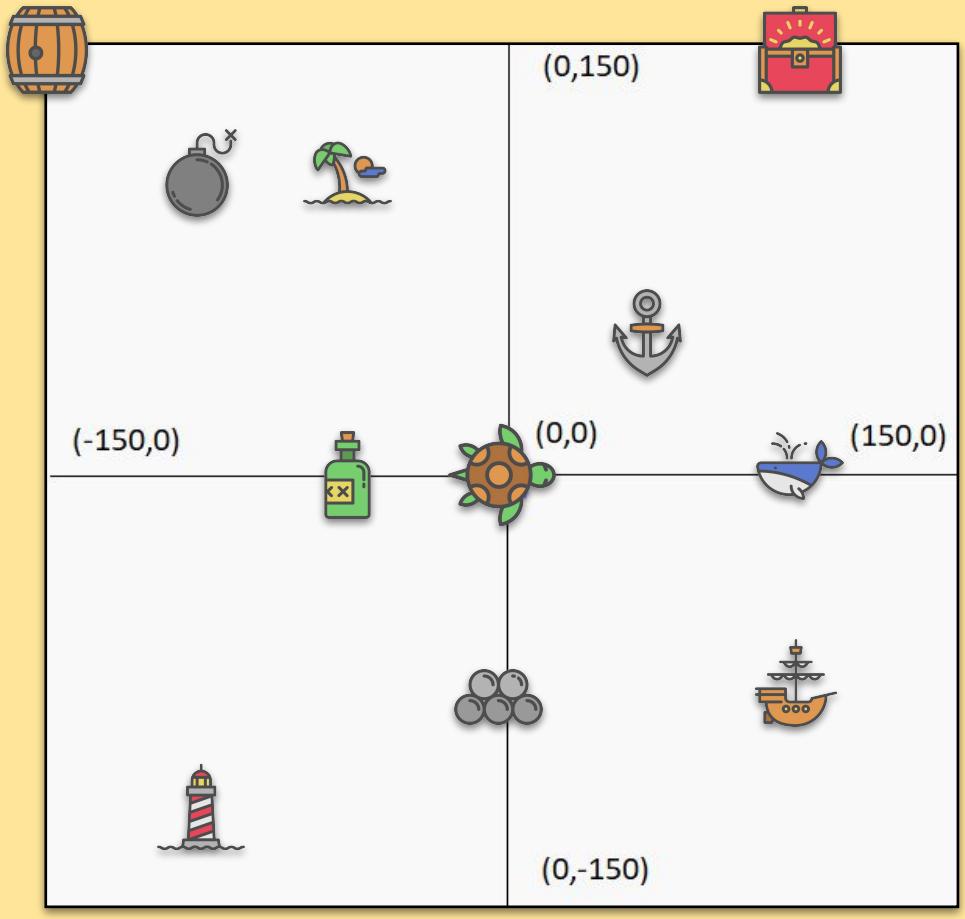
- **Code Combat (above)**
  - www.codecombat.com/play/dungeon

Drawing Location - coordinates

Sometimes we may wish to move the turtle before we start drawing something so we can draw shapes in different locations on the screen.

Drawing Location - Treasure Hunt

Remember this map? We used it for Scratch before. As a class, try and guess the approximate **coordinates** of each of the items of treasure for Tina.



Barrel of Oil
(-150,_____)

Bottle of Poison
(-50,_____)

Whale
(100,_____)

Lighthouse
(-100,_____)

Ship
(____, -75)

Desert Island
(____,____)

Cannonballs
(____,____)

Bomb
(____,____)

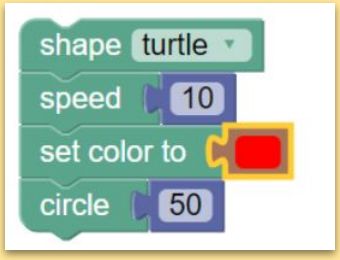
Treasure
(____, 150)

Anchor
(____,____)

Task 1 - Drawing Two Circles

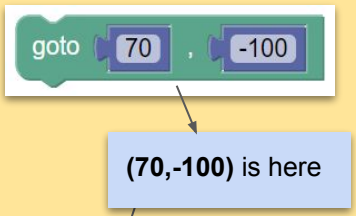
www.bit.ly/blankblocktrinket

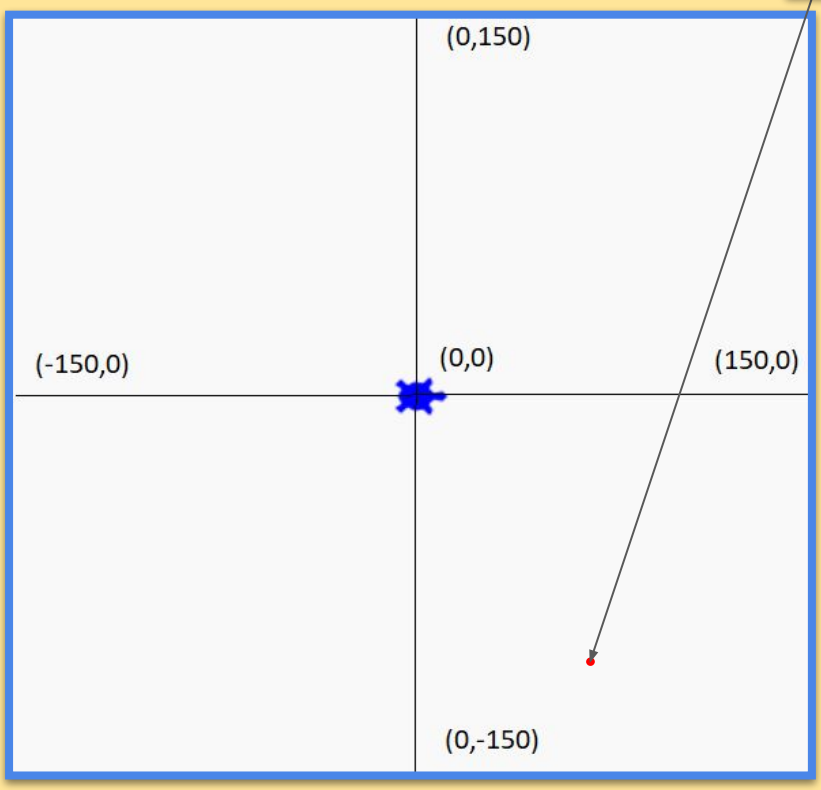
1. Set up Tina the Turtle as before being sure to include **shape, speed, colour** and **circle.**



2. Now we want to move her to a different location. We need to use **Pen up** to make sure Tina does not draw lines while moving to a different location.

3. Tell Tina to go to a particular point on the screen by telling her the **x-coordinate** (horizontal/ left right) and the **y-coordinate** (vertical/up-down)



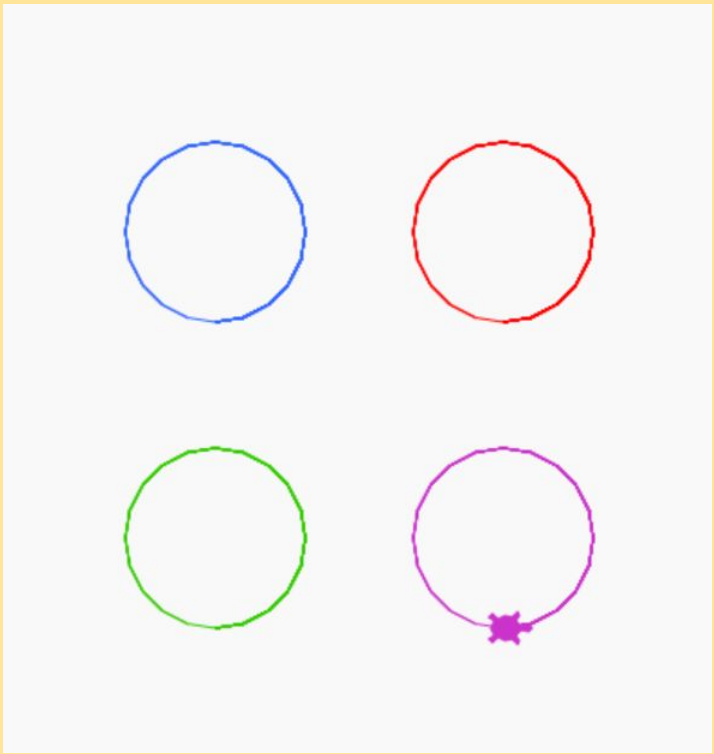


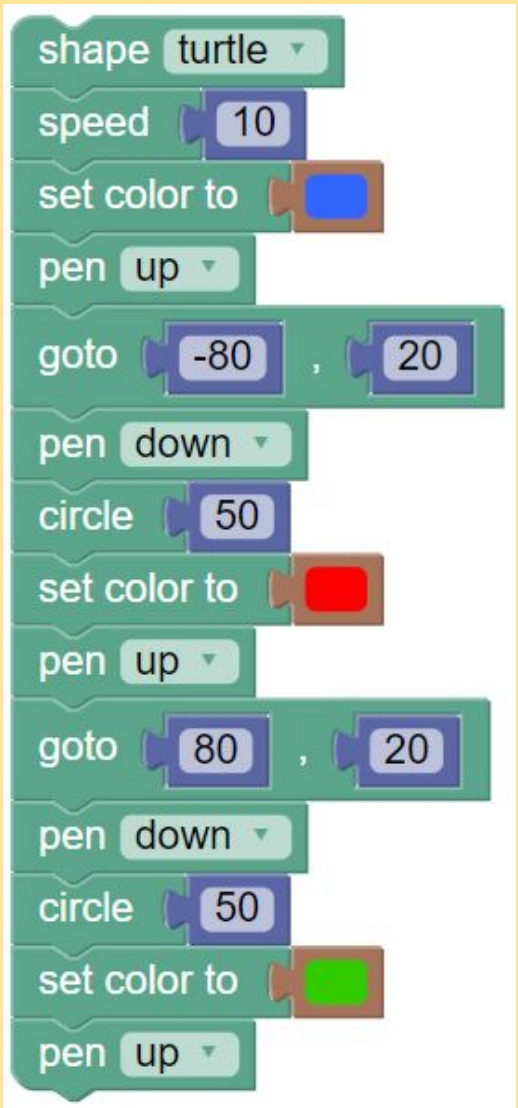
4. Put the pen back down on the screen using **Pen Down** so that Tina can start drawing again.

5. Start drawing your shape.

Task 2 - Drawing Four Circles

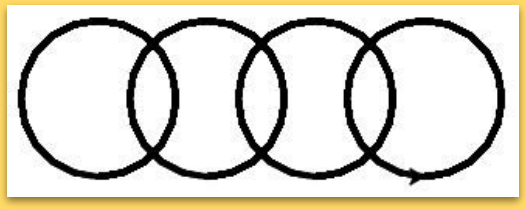
Finish the code on the right so that it makes four circles similar to the drawing below.

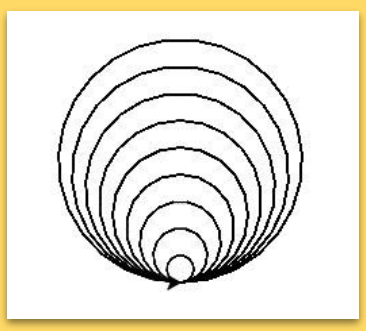


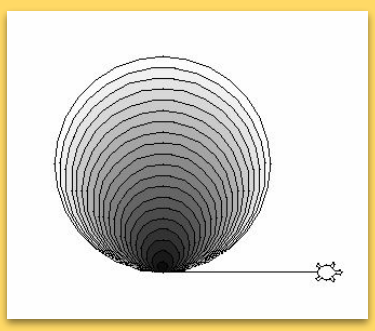


Challenge

Try making the image these images.

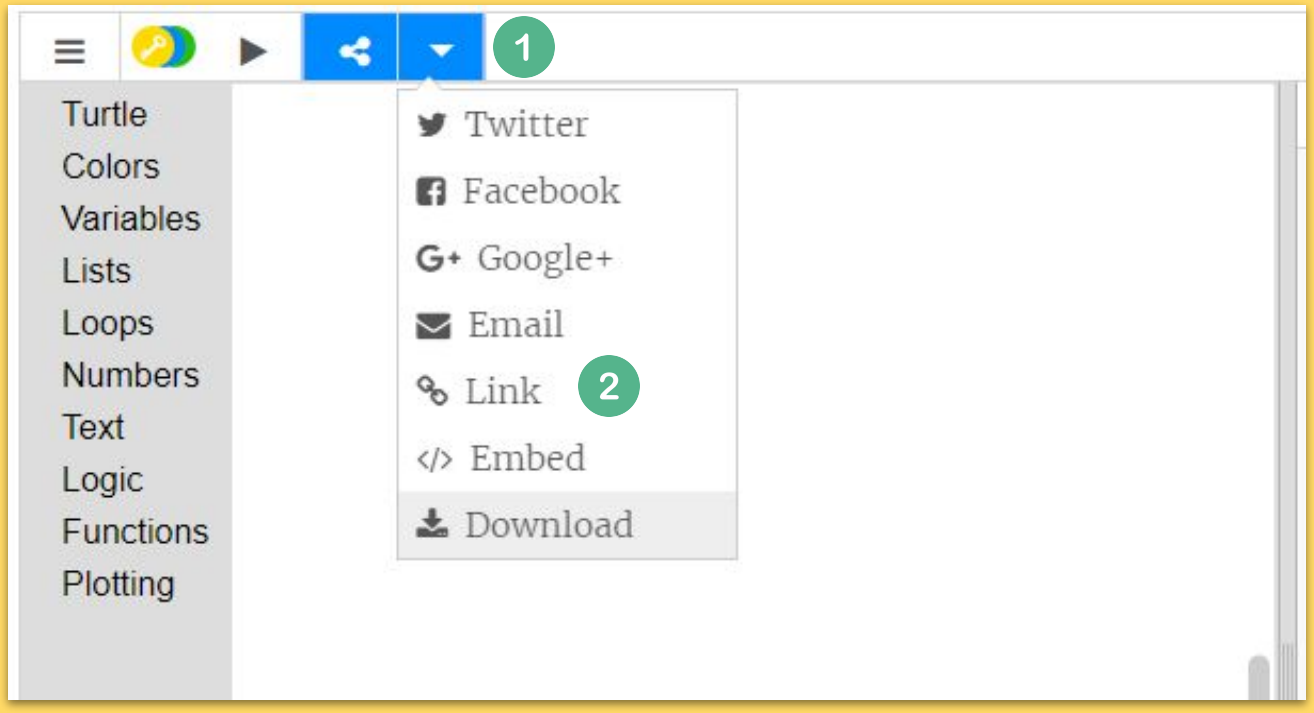






Now it's time to Save our Work

Make sure you do this properly, **you'll need it next week!**



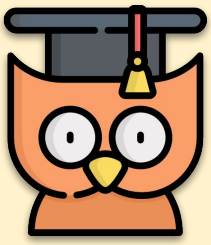
To save your Python project:

1. Open the drop down menu.
2. Click **Link**
3. Paste the Link into Notepad and save to your USB
   If you're not sure to do any of this, **ask your tutor.**

Extra Challenge

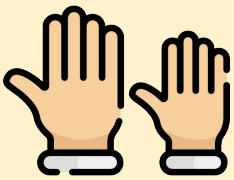
If you're teacher is happy with your work and when you've saved it properly, play the ***Basics*** section (Part 1) of Lightbot at this LINK.

Lesson 4 - Block Turtle III



Learning Outcomes:

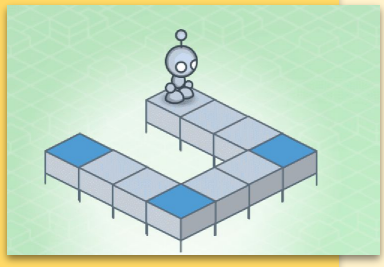
- Making Tina draw some simple shapes at different locations on the map - Olympics Rings
- Making a miniature house
- Make other shapes with Tina



REMEMBER: Put up your hand. We love to help!

Game Time

Today we're going to look at how to draw different shapes apart from just circles. Before we look at how to draw these shapes we're going to play Lightbot which can be accessed at www.lightbot.com/flash.html. The concepts we will learn in Lightbot will link in nicely with drawing some of these shapes.



Refresher Task - Multiple Circles

Lets see if you can remember some of the things we learned about last week (it's ok to look back!). As a test, make **multiple circles** in various colours and positions. If you are stuck, open last weeks work. You will need:

- Shape 

- Speed 

- Colour 

- Pen Up 

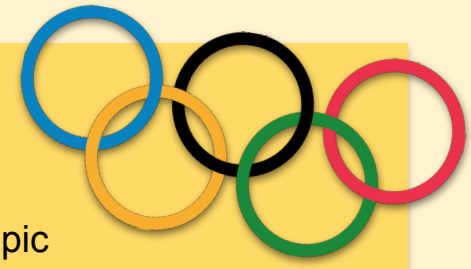
- Go to 

- Pen Down 

- Circle 

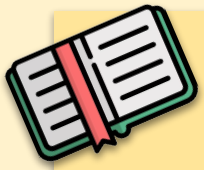
http://www.taoc.ie/turtle2

Must-Do Challenge



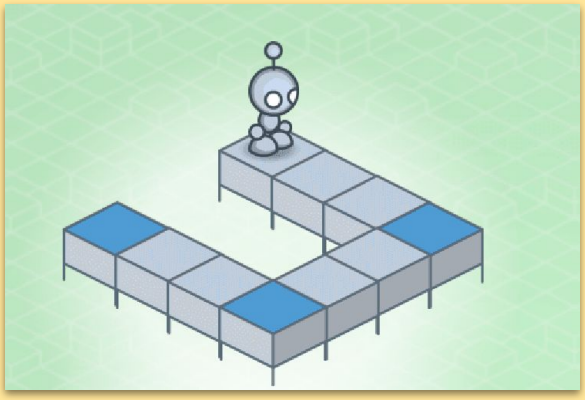
This challenge should be a tricky one. Imagine *The Olympic Council of Ireland* have asked you to draw the Olympic flag with Python so they can use it on their new website.

Use what you have learned in this lesson to write code to make Tina the Turtle draw the Olympic flag. https://taoc.ie/olympicstarter



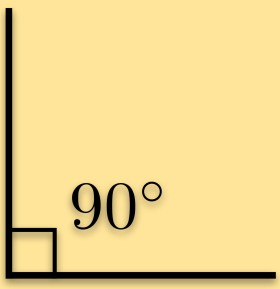
Drawing Squares

We can draw shapes other than circles. Although drawing other shapes isn't just as easy. Think about how you would make *Lightbot* move to make a square.



How would we do this?

Drawing a square with Tina must be done in a similar way. Drawing 4 separate straight lines and with four 90 degree angles.

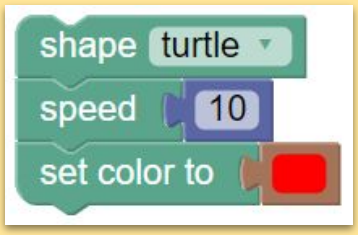


Let's get coding!

www.bit.ly/blankblocktrinket

We are now going to make Tina draw a square.

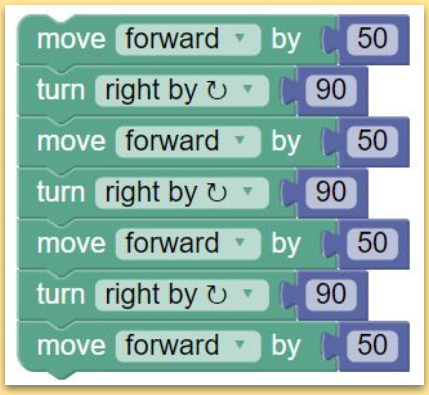
1. Set up Tina like we did in the last lesson (set color, set speed etc.).

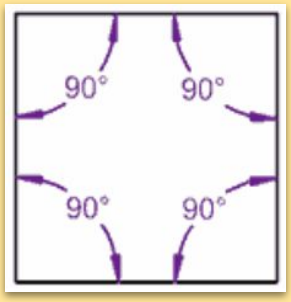


Expert Tip

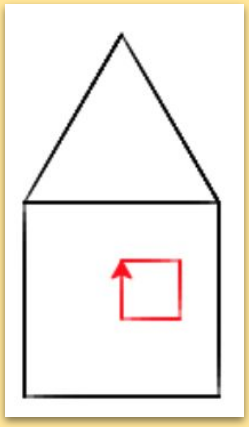
Be careful where you put the **fill** or **color** commands. You will get different results depending on where you put them.

2 Now Tell Tina to do the following code





3 Try drawing a triangle for a roof and another square for the window to make a house



4 Using the **begin fill** and **end fill** commands, colour the shapes

Other Shapes

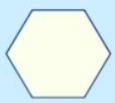
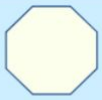
Have a try and see if you can finish some other shapes based on what you've learned about drawing a square.

Expert Tip

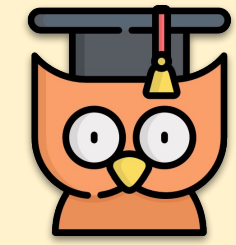
The angle which Tina turns isn't the degrees of each angle. The **Turn Angle** or **Outside Angle** is calculated by taking the inside angle away from 180°.

eg Pentagon is 180° - 108° = **60°**

Hexagon is 180° - 60° = **120°**

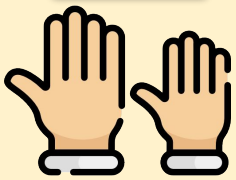
Sides	Shape	Each Angle	Turn Angle 180°- Angle
3		60°	120°
4		90°	90°
5		108°	72°
6		120°	60°
8		135°	45°
9		140°	40°

Lesson 5 - From Blocks to Code I



Learning Outcomes:

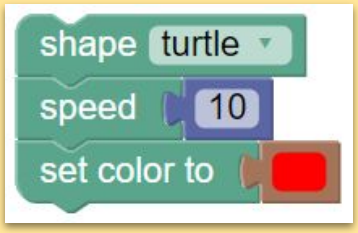
- Learning how to make text with Tina the Turtle
- Make a pizza for Fat Tony's Pizza company
- Converting block python into typed python code (Grade 2)



N.B. Have you washed your hands before getting started?

Writing with Turtles

Other than drawing lines and shapes, Tina the Turtle can also write words and sentences onto the canvas. We are now going to write something onto the canvas using Tina the Turtle. To practice this open up a blank block trinket: www.bit.ly/blankblocktrinket. **Grade 3** - start with these blocks and click ***View Code*** and click ***Convert to Python***.



1. Set up Tina the Turtle
   Make turtle, set color, set speed etc.
2. Tell Tina what to write on the canvas.

3. Run the code above.

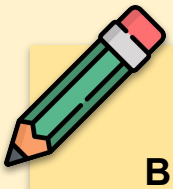
```
turtle.write('Hello World!', None, None, "18pt Impact")
```

4. Change the color of the font.
5. Change the font size.

Grade 3 - Expert Tip (What the line means)

```
turtle.write('Hello World!', None, None, "18pt Impact")
```

Using the write command from the turtle library	'Your Message'	Move *true/false*	Align *left/ center/ right*	Font Size and Family

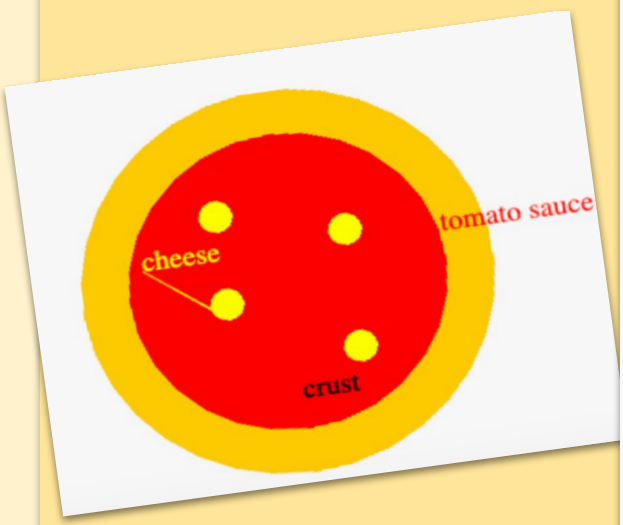


Your Main Task

Block Version: https://taoc.ie/pythonpizzastarter
Code Version: https://repl.it/@DamienAOC/Python-Pizza

Fat Tony, a sweaty, overweight, middle-aged pizzeria owner wants you to come up with a new pizza recipe to lure in new customers. Use **Text** along with a picture of the pizza.
Use what you have learned so far to write code to make Tina the Turtle draw a delicious new pizza design with your own favourite toppings.

Hint: Once you have drawn the **outer crust** and **base** of the pizza, draw **different shapes** with **different colours** to draw your different toppings.



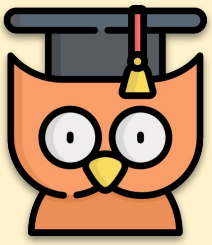
Some examples from our students in **St. Conleth's Junior School**

Finished Early?

Try recreating some of the advanced art cards at:

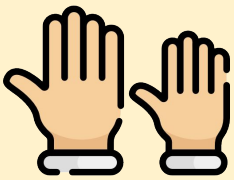
https://wiki.sugarlabs.org/go/Activities/Turtle_Art/Turtle_Cards

Lesson 6 - From Blocks to Code II



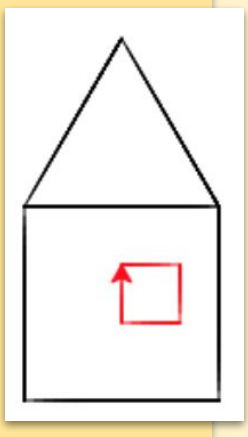
Learning Outcomes:

- Revising Shapes and introducing **For Loops**
- Make a house scene with as many items as possible with the view to using different shapes



REMEMBER: Put up your hand. We love to help!

Revision - Shapes and Angles



A few lessons ago we asked you to try and make a miniature house similar to the one on the right. Today we're going to expand on this to make a **house scene** that contains a list of essential items.

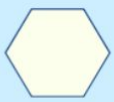
Click on one of the links below and for practice draw **and fill** each one of the shapes on the right.

Block Version:

https://taoc.ie/turtle4

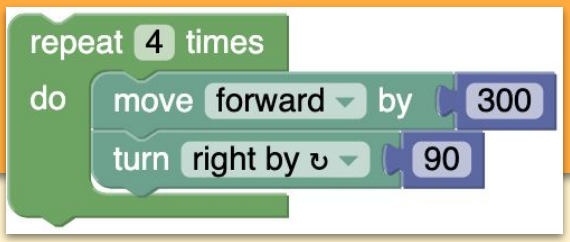
Code Version:

https://taoc.ie/replturtle4

Sides	Shape	Each Angle	Turn Angle 180°- Angle
3		60°	120°
4		90°	90°
5		108°	72°
6		120°	60°
8		135°	45°
9		140°	40°

Want to save time? Use For Loops

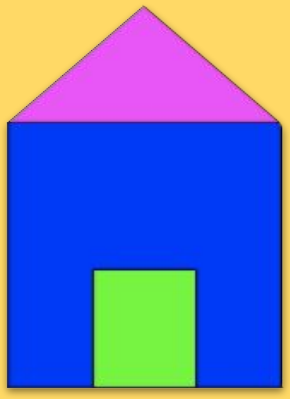
They can help us save time here. Instead of using many lines of code to draw a square for example, we could just say:



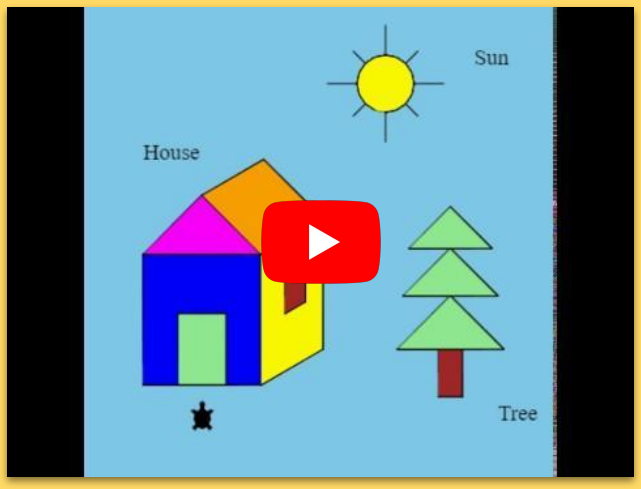
House Scene Challenge

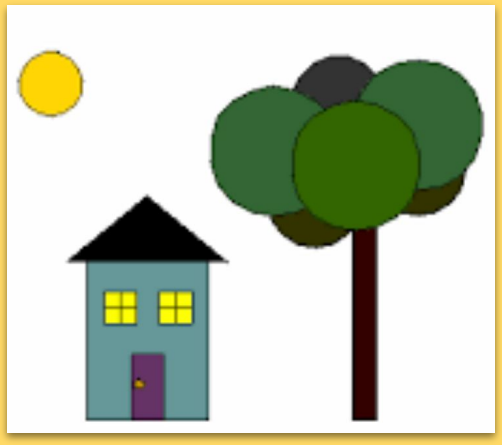
Block Version: https://trinket.io/embed/blocks/563fe95f6b
Code Version: https://repl.it/@DamienAOC/Python-House

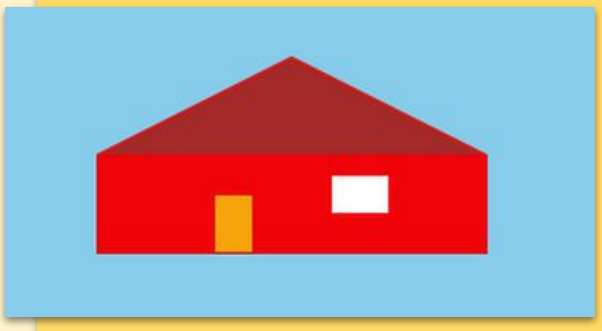
Your challenge is to use everything you've earned to make a house. Around this you could have a road, a car, a UFO or whatever you feel like but your scene **MUST HAVE** the following:

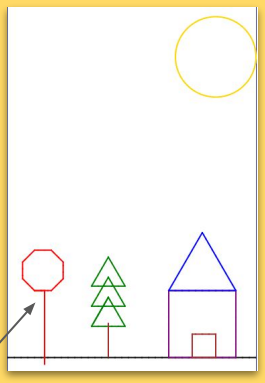


- a **house** with a roof, door and windows
- a **tree** (a rectangle with circles or triangles)
- a **sun** (a circle)
- **Stop sign** (a line/rectangle with a **red octagon** on top as in the second example on the second row below)









Stop Sign

Extra Challenge

Try recreating some of the advanced art cards at:

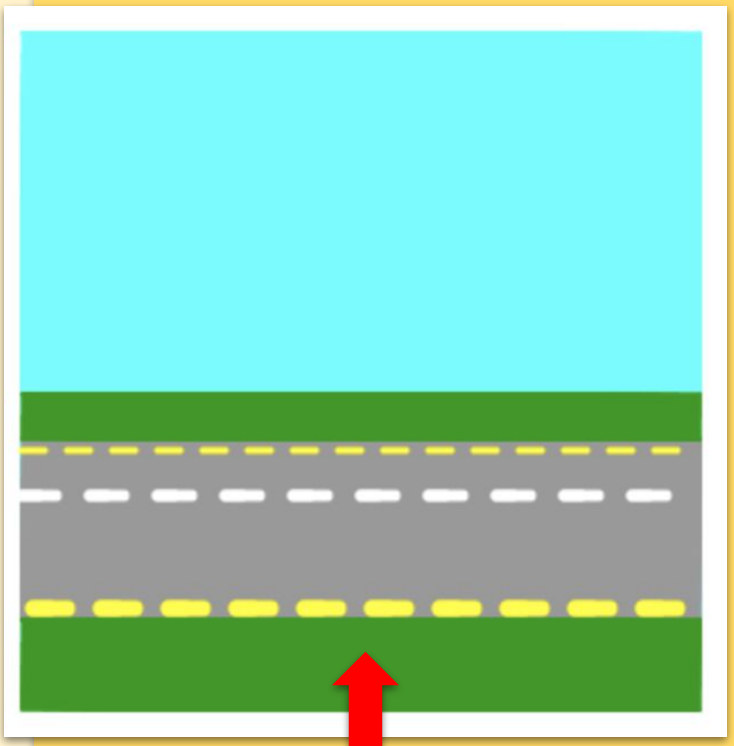
https://wiki.sugarlabs.org/go/Activities/Turtle_Art/Turtle_Cards

Traffic Scene (to be worked on over two lessons/ for homework)

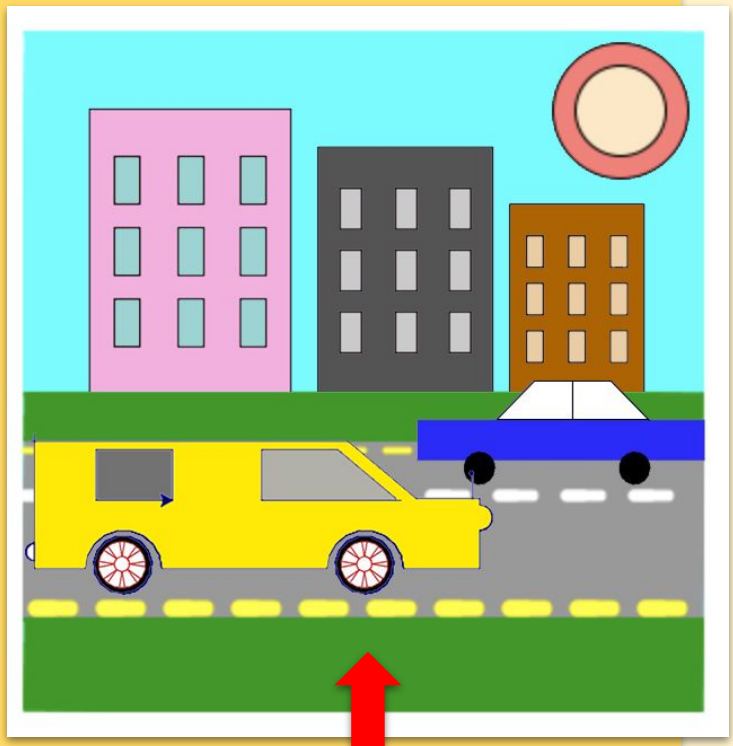
Block Version: https://taoc.ie/trafficstarter (Only use if you really need)

Code Version: https://taoc.ie/repltrafficstarter

Here a scene you'll most likely need to **save** to work on over two lessons. Have a look at this digital artist's **HUGE AND AMAZING** code of a 3D city scene, made using Turtle Graphics. This isn't what we're looking for, but it may give you some inspiration.



Your Starting Code



What you do is up to you!

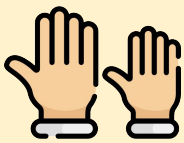
Start by trying to draw a car or maybe a skyscraper. Then after that, it's up to you!

NB: If you intend to do a big scene **or** even if you just feel like being lazy and calling upon your two best mates 'Copy' and 'Paste' to save you time (find me someone who doesn't?!) **PLEASE PLEASE PLEASE USE THE CODE VERSION** instead of the block version. It will save you so much time and hassle and you'll be coding like a pro! YOU CAN DO THIS!!

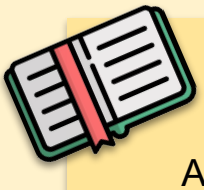
Lesson 7 - From Blocks to Code III

Learning Outcomes:

- To Learn the importance of For Loops and Variables
- Making a Shape Maker App
- Letting Python Do the Maths

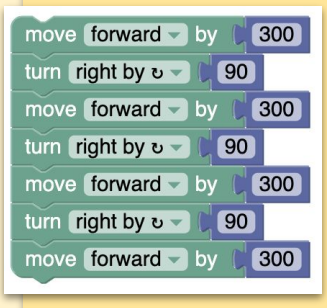


REMEMBER: Put up your hand. We love to help!

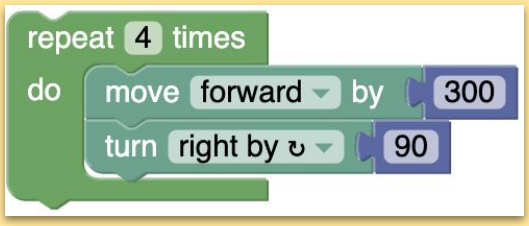


For Loops and Variables

As mentioned before, loops are a super important concept in coding.. A **"for" loop** completes a task a set number of times, then completes and exits.



Bad Code

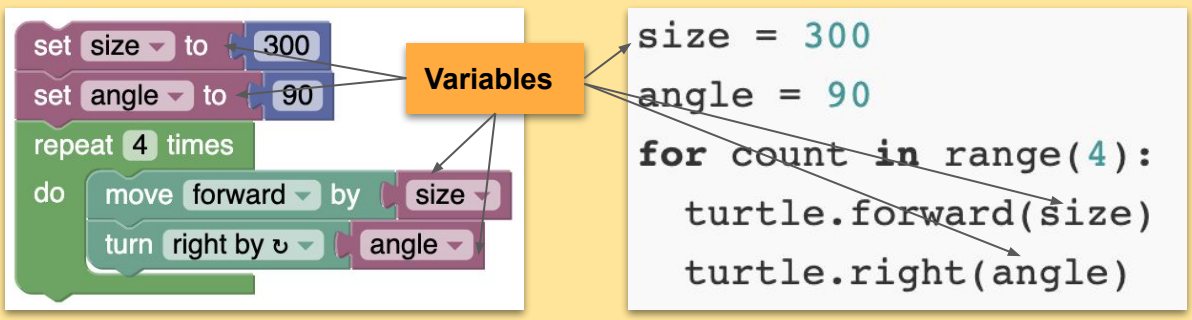


Better Code (Less work!)

```
for count in range(4):
  turtle.forward(300)
  turtle.right(90)
```

The 'Real' python code

Students who have used *Scratch* before will know **variables** as being a container of data, data which can change in a programme - score, bullets left, X-position, the amount of coins collected etc. If we wanted the above shape to change when the user wants it to, we could use variables.



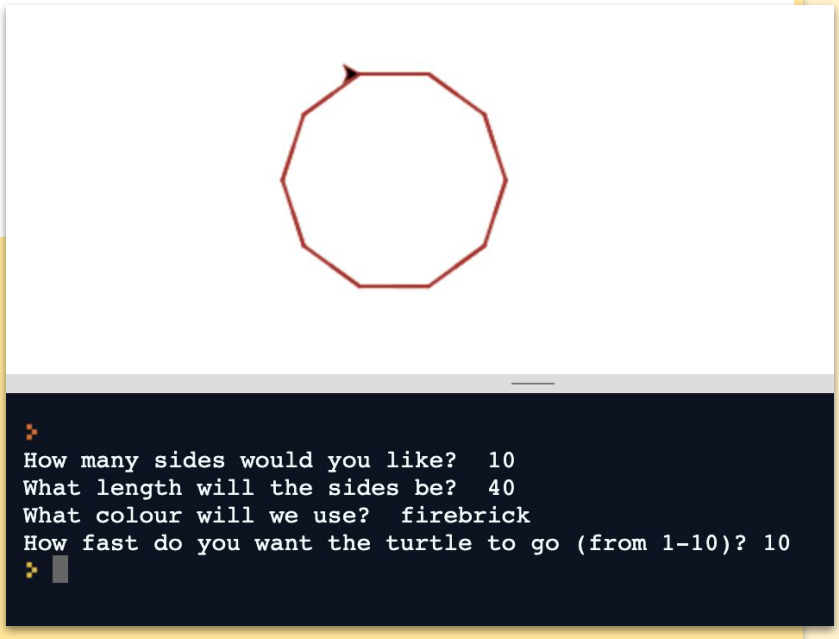
Incase you haven't notices, **ALL** codes on this page do this exact same thing, but some are more professional than others.

Making a Shape Maker App

Starting code: https://taoc.ie/shapemakerstarter

Your task is to turn the starting code into the code below. The app has 5 variables, **sides, length, colour, speed** (values all determined by the user) as well as **angles** (which the computer calculates itself by dividing 360 by 'sides)'.

```
1  import turtle
2  turtle.shape(turtle)
3  turtle.pensize(3)
4
5  #The App asks the user to INPUT information
6  sides = int(input("How many sides would you like? "))
7  length = int(input("What length will the sides be? "))
8  colour = input("What colour will we use? ")
9  speed = int(input("How fast do you want the turtle to go (from 1-10)?"))
10 angles = 360/sides
11
12 #Here the app draws whatever shape the user wants
13 for i in range(sides):
14     turtle.speed(speed)
15     turtle.pencolor(colour)
16     turtle.forward (length)
17     turtle.right(angles)
18
```



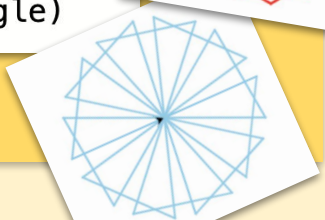
This code if typed correctly should make a solid, closed shape each time. Be careful putting in ridiculous numbers. Long variables **WILL** make your programme crash and you may lose your work.

Extra Challenge

Develop the code on the right to make a Custom Shape Spirograph Making app. The variable `turnangle` will need to be another question.

NB Watch out for those tabbed spaces in the for loop!

```
for i in range(20):
    for i in range(sides):
        turtle.speed(speed)
        turtle.pencolor(colour)
        turtle.forward (length)
        turtle.right(angles)
    turtle.right(turnangle)
```



Letting Python do the Maths

People who don't know much about code often think that you need to be good at maths to be good at coding. However, smart coders get computers to do the maths for them! We're going to let Python work out **how old a user will be in 2100.**

1. **CTRL+CLICK** https://taoc.ie/blankpython and type in the code as it appears here. Instead of 2009, **put in your birth year**).

```
1  #!/bin/python3
2
3  print(2100-2009)
```

2. Improve your program by using input() to ask the user their age and store it in a **variable** called born.

```
born = input('What year were you born?')
print(2100-born)
```

3. *R*un your program and then enter the year you were born. Did you get another error message? That's because anything typed into your program is **text**, and it needs to be converted to a **number**. You can use int() to convert the text to an **integer** ('integer' means 'whole number' like 5, 929 or 1,000,000 **not** 2.1 or 5.65 etc).

```
born = input('What year were you born?')
born = int(born)
print(2100-born)
```

Adding another Variable

4. So far we used a variable called **born**, and we asked a question that generates a number. To make the programme make more sense to the user, change your code to **exactly** as it appears below. But first, think about what it does.

```
#!/bin/python3

born = input('What year were you born?')
born = int(born)
age = 2100 - born
print('In the year 2100, you will be',age,'years old!')
```

```
Powered by trinket
What year were you born? 2009
In the year 2100, you will be 91 years old!
```

Extra Challenge - Dog Years

Your challenge is to write a programme that asks users their age and then tells them their age in dog years.

```
What is your age? 9
If you were a dog, you'd be 63 !!

 o____
  ||||
```

Expert Tip

In python, the symbol for **multiplication** is *, and is usually **shift+8** on the keyboard.

Extra Challenge - Art Challenge

Work out what this code does.

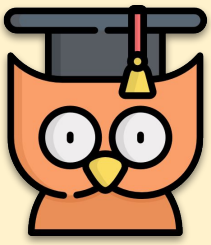
```
print('ha ' *4)
print('ba' + 'na' *2)
print('Hello' + '!' * 10)
```

```
print('Here is a scarf:')
print('~#' * 10)
print('#-' * 10)
print('Here is a wave:')
print('/\  ' * 10)
print('  \/' * 10)
```

Now type out this code. Can you make any word drawings or patterns of your own?

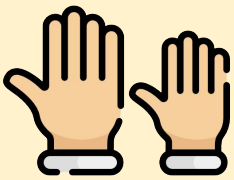
```
Here is a scarf:
~#~#~#~#~#~#~#~#~#~#
#-#-#-#-#-#-#-#-#-#-
Here is a wave:
/\  /\  /\  /\  /\  /\  /\  /\  /\  /\
  \/  \/  \/  \/  \/  \/  \/  \/  \/  \/
```

Lesson 8 - Python II - Rock Paper Scissors



Learning Outcomes:

- Creating a rock/paper/ scissors game
- Using simple if statements
- Announcing the winner



REMEMBER: Put up your hand. We love to help!

Let's Get Coding

CTRL+CLICK **repl - https://taoc.ie/turtleracestarter**

trinket - https://taoc.ie/blankpythontrinket

1 This project contains these words.

```
from random import randint
```

We'll use them later to generate random numbers.

2 First, we're going to let the player choose Rock, Paper or Scissors by typing **r, p or s.** Add the blue lines of code. The second will print the player's choice.

```
from random import randint

player = input('rock (r), paper (p) or scissors (s)?')

print(player, 'vs')
```

3 Now to code the computers guess. randit will generate a random number to decide between rock, paper and scissors. Add in the blue lines and test it a few times to see that you're getting a random number.

```
player = input('rock (r), paper (p) or scissors (s)?')

print(player, 'vs')

chosen = randint(1,3)
print(chosen)
```

Coding the Computer

In our code, **1 = rock (r), 2 = paper (p)** and **3 = scissors(s).** We're going to use an **if statement** to check **if** the number 1 (for example) is randomly chosen, the computer will see that as a rock. Add the lines in blue making sure to **indent** (move to the right). We can use two spaces or a 'tab' for this. Tab is usually above the **Caps Lock** button on your keyboard.

```
chosen = randint(1,3)
print(chosen)
if chosen == 1:
  computer = 'r'
```

Don't forget the colon ':'

Two spaces or a 'tab'

Short for "else if"

```
chosen = randint(1,3)
#print(chosen)
if chosen == 1:
  computer = 'r'
elif chosen == 2:
  computer = 'p'
else:
  computer = 's'
print(computer)
```

Comment out this line by putting a '#' at the start

Add the rest of the code on the left, but first, *what does it do?* When finished, test your code to see if it works.

The computer's choice gets printed on a new line. You can fix that by adding `end=' '` after `vs`. This tells Python to end with a space instead of a new line. Play the game a few times by clicking Run and making a choice. For now you'll have to work out who won yourself. Next you'll add the Python code to work this out.

```
print(player, 'vs', end=' ')

chosen = randint(1,3)
#print(chosen)
```

```
rock (r), paper (p) or
scissors (s)? p
p vs r
```

Announcing the Winner

Lets look at a case where the player choses ‘r’ (rock). If the computer chose ‘s’ (scissors) then the player wins (rock beats scissors). If the computer chose ‘p’ (paper) then the computer wins (paper beats rock). We can check the player choice *and* the computer choice using the word **and**.

```
if player == computer:
  print('DRAW!')

elif player == 'r' and computer == 's':
  print('Player wins!')

elif player == 'r' and computer == 'p':
  print('Computer wins!')
```

Add in more elif’s so that every ‘p’ and ‘s’ combination is covered.

Challenge: ASCII Art

Instead of using the letters r, p and s to represent rock, paper and scissors, can you use ASCII art? For example:

```
rock (r), paper (p) or scissors
(s)? s
>8 vs ___
Player wins!
```

Challenge: Create a new game

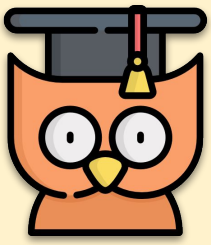
Can you create your own game like Rock, Paper, Scissors with different objects?

Click the ‘Duplicate’ button to make a copy of your Rock, Paper Scissors project to start from.

This example uses Fire, Logs and Water:

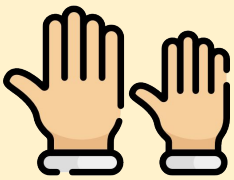
```
Fire, Logs, Water
Fire burns Logs
Logs make a bridge over Water.
Water puts out Fire
fire (f), logs (l) or water (w)? l
@@@ vs ~~~
Player wins!
```

Lesson 9 - Python III - Turtle Race



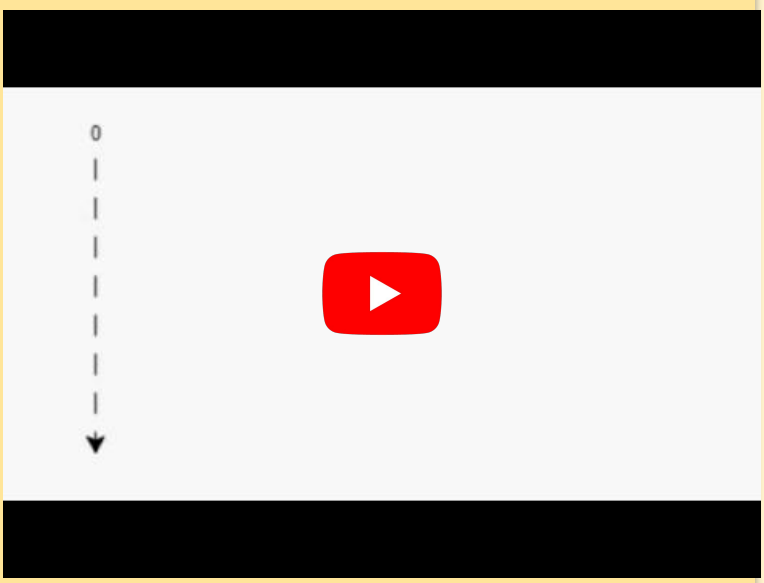
Learning Outcomes:

- Make a random race which races turtles for fun
- Learning about for loops
- Customising your game



REMEMBER: Put up your hand. We love to help!

Let's Get Coding



1. **CTRL+CLICK** this video to see the end result. The game will be similar to a horse racing game you may have seen before.

2. **CTRL+CLICK** this link: https://taoc.ie/turtleracestarter https://taoc.ie/blankpy **(trinket)**

3. Write this code and see what happens. Then change the code so that it makes a pattern from **1-5** using the pattern below:

```
#!/bin/python3
from turtle import *

write(0)
forward(100)
write(5)

```

```
#!/bin/python3
from turtle import *

write(0)
forward(20)
write(1)
forward(20)
write(2)
forward(20)
write(3)

```

On the next page we'll discuss **for** loops, a method of making writing this long code easier, but for now, copy the code on the right.

For Loops - Making it easy

1 The code we wrote is very repetitive. There is an easier way to code this in python called a **for** loop. Change your code so it looks like the code on the right. We need to give it a **range of 6** because there are **6** numbers between 0 and 5.

```
#!/bin/python3
from turtle import *

for step in range(6):
  write(step)
  forward(20)
```

```
#!/bin/python3
from turtle import *

penup()
goto(-140,140)

for step in range(15):
  write(step)
  forward(20)
```

2 Expand the code so that it matches what appears on the left.

But first, what does it do?

3 To draw the lines, we're going to have to turn the turtle to the right, before putting a pen down, drawing a line of 150, picking the penup and moving backward to the start of the line. Add the code in blue below and see what happens.

```
for step in range(15):
  write(step, align='center')
  right(90)
  forward(10)
  pendown()
  forward(150)
  penup()
  backward(160)
  left(90)
  forward(20)
```

`right(90)` makes the turtle turn right 90 degrees (a right angle.) Moving **`forward(10)`** before putting the pen down leaves a small gap between the number and the start of the line. After drawing the line you lift up the pen and go **`backward(160)`** the length of the line plus the gap.

Racing Turtles

1 When you use commands like **`forward(20)`** you are using a single turtle. But you can create more turtles. Add the following code to the end of your script (but make sure it's not indented)!

```
redTurtle = Turtle()
redTurtle.color('red')
redTurtle.shape('turtle')
redTurtle.penup()
redTurtle.goto(-160,100)
redTurtle.pendown()
```

The first line creates a turtle called 'red'. The next lines set the colour and shape of the turtle. Now it really looks like a turtle!

Now you need to make the turtle race by moving a **random number of steps** at a time. You'll need the **`randint`** function from the Python `random` library. Add this `import` line to the top of your script. The randint function returns a random number between the values chosen. The turtle will move forward **1, 2, 3, 4, or 5** steps each turn.

2

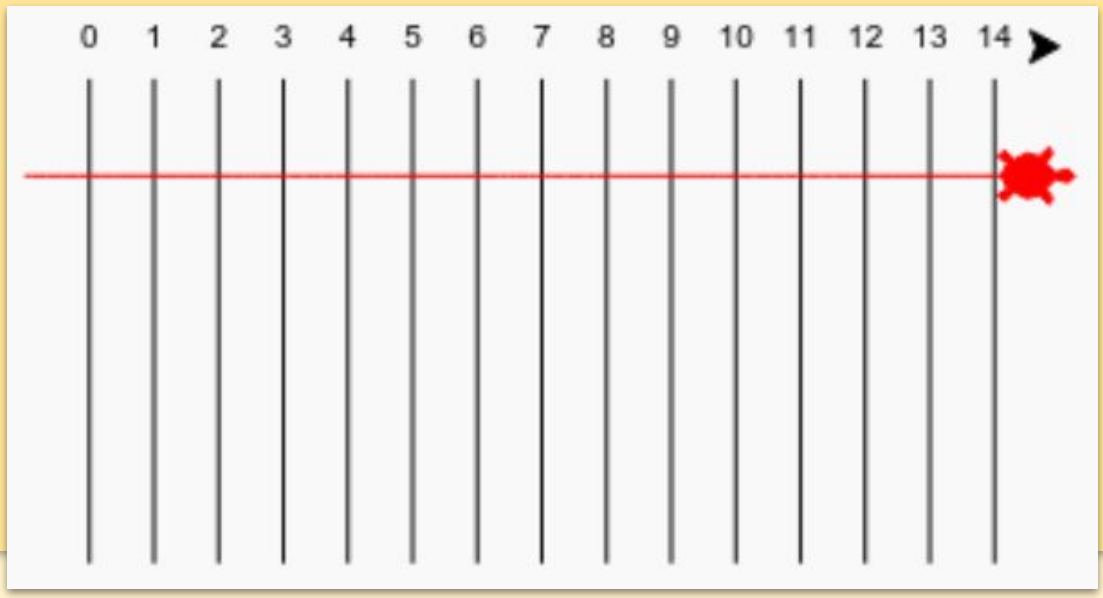
```
from turtle import *
from random import randint
```

```
redTurtle.goto(-160,100)
redTurtle.pendown()

for turn in range(100):
  redTurtle.forward(randint(1,5))
```

3

Test the code and you should get a one-turtle race! We'll add more next.



Adding more Turtles

```
redTurtle = Turtle()
redTurtle.color('red')
redTurtle.shape('turtle')
redTurtle.penup()
redTurtle.goto(-160,100)
redTurtle.pendown()

blueTurtle = Turtle()
blueTurtle.color('blue')
blueTurtle.shape('turtle')
blueTurtle.penup()
blueTurtle.goto(-160,70)
blueTurtle.pendown()

greenTurtle = Turtle()
greenTurtle.color('green')
greenTurtle.shape('turtle')
greenTurtle.penup()
greenTurtle.goto(-160,40)
greenTurtle.pendown()

yellowTurtle = Turtle()
yellowTurtle.color('yellow')
yellowTurtle.shape('turtle')
yellowTurtle.penup()
yellowTurtle.goto(-160,10)
yellowTurtle.pendown()

for turn in range(100):
  redTurtle.forward(randint(1,5))
  blueTurtle.forward(randint(1,5))
  greenTurtle.forward(randint(1,5))
  yellowTurtle.forward(randint(1,5))
```

Copy

Paste

Paste

Paste

Copy

Paste

Paste

Paste

A one turtle race isn't much of a game! Add the code on the left to get the remaining turtles going.

You should be able to see a pattern here so don't forget...

REMEMBER TO:

COPY (*CTRL + C*)

AND

PASTE (*CTRL + V*)

Copy and Paste are a coders best friends!

Challenge: Do a twirl

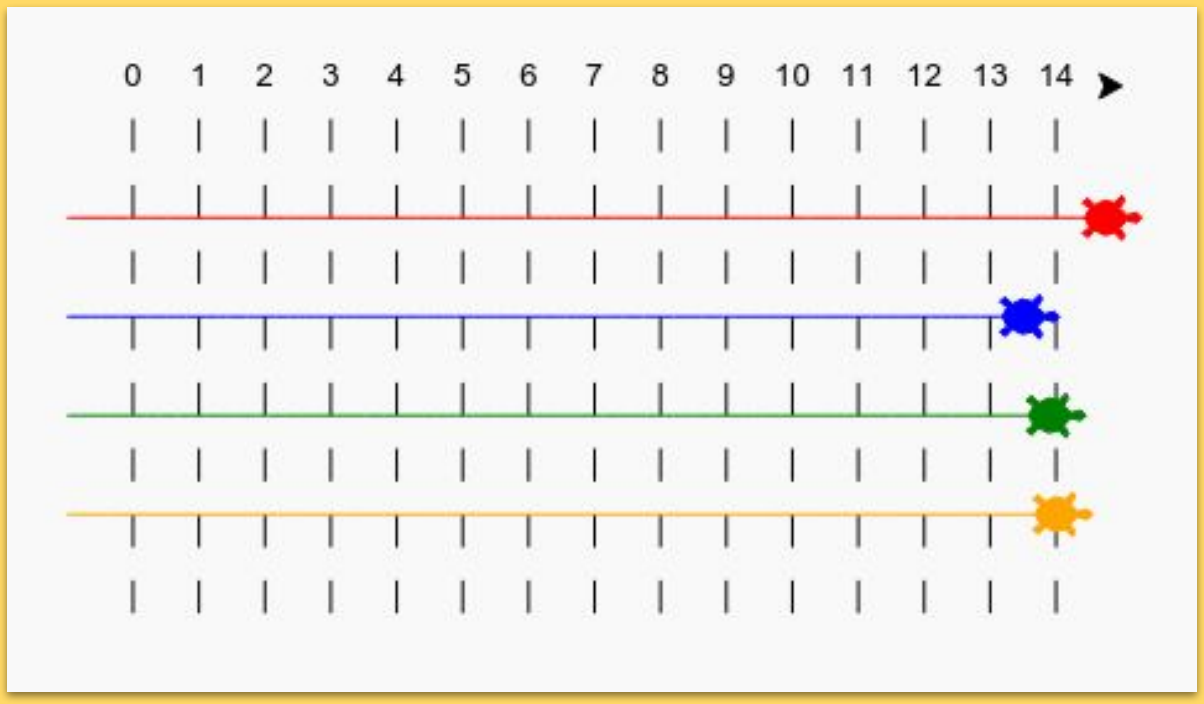
Can you use a **`for turn in range():`** loop to make each turtle do a 360 degree twirl after they get to the starting line? You'll need to make sure they are facing in the right direction at the start of the race!

`redTurtle.right(36)` will turn the red turtle right by 36 degrees.

Hint: A full turn is 360 degrees. A turtle could turn right 10 degrees 36 times, or left 5 degrees 72 times, or any other numbers make 360!

Challenge: Dashed lines

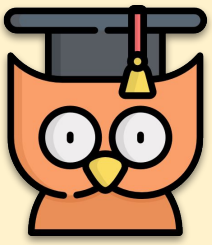
Can you use a loop to make the track lines dashed instead of solid?



Hint: Find the code that draws a straight line.

Try using:`for`, `forward()`, `penup()` and `pendown()`

Lesson 10 - Python Project IV - ISS

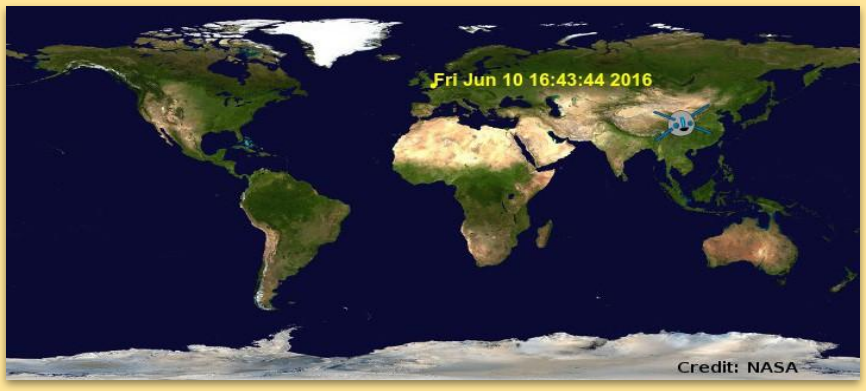


Learning Outcomes:

- Using Web Service to find the location of the International Space Station
- Plotting its location on a map.

Introduction

In this project you will use a **web service** to find out the **current location** of the International Space Station (ISS) and **plot its location** on a map.



Step 1: Who is in Space?

You're going to use a **web service** that provides live information about space. First let's find out who is currently in space. A web service has an address (url) just like a web page does. Instead of returning HTML for a web page it returns data.

1. Open **http://api.open-notify.org/astros.json** in a **web browser** (Chrome, Internet Explorer, Firefox etc.). You should see something like what is here (the data is live so your figures will be different). The data format is called **JSON** (pronounced Jay-Son).


```
{
 "message":
 "success",
 "number": 3,
 "people": [
  {
   "craft": "ISS",
   "name": "Yuri Malenchenko"
  },
  {
   "craft": "ISS",
   "name": "Timothy Kopra"
  },
  {
   "craft": "ISS",
   "name": "Timothy Peake"
  }
 ]
}
```

2 Let's call the web service from Python so we can use the results. Open this project file: www.taoc.ie/issrepl **(repl)** or www.taoc.ie/iss **(trinket).** The **urllib.request** and **json** modules have already been imported for you.

3 Add the following code to **main.py** to put the web address you just used into a variable: `url = 'http://api.open-notify.org/astros.json'`

4 Now let's call the web service:

```
url = 'http://api.open-notify.org/astros.json'
response = urllib.request.urlopen(url)
```

5 Next you need to load the JSON response into a Python data structure:

```
url = 'http://api.open-notify.org/astros.json'
response = urllib.request.urlopen(url)
result = json.loads(response.read())
print(result)
```

You should see something like this.

```
{'message': 'success', 'number': 3, 'people': [{'craft': 'ISS', 'name': 'Yuri Malenchenko'}, {'craft': 'ISS', 'name': 'Timothy Kopra'}, {'craft': 'ISS', 'name': 'Timothy Peake'}]}
```

This is a Python dictionary with 3 keys: **message, number and people.**
The 'success' value of message tells you that the request was successful.

6 Now let's print the information in a more readable way. First, let's look up the number of people in space and print it:

```
url = 'http://api.open-notify.org/astros.json'
response = urllib.request.urlopen(url)
result = json.loads(response.read())

print('People in Space: ', result['number'])
```

`result['number']` will print the value associated with the key 'number' in the result dictionary. In the example this is 3. The value associated with the 'people' key is a list of dictionaries! Let's put that value into a variable so you can use it.

```
print('People in Space: ', result['number'])

people = result['people']
print(people)
```

You should see something like this:

```
[{'craft': 'ISS', 'name': 'Yuri Malenchenko'}, {'craft': 'ISS', 'name': 'Timothy Kopra'}, {'craft': 'ISS', 'name': 'Timothy Peake'}]
```

Now you need to **print out a line** for each astronaut.

You can use a **for loop** to do this in Python. Each time through the loop p will be set to a dictionary for a different astronaut.

```
print('People in Space: ', result['number'])

people = result['people']

for p in people:
  print(p)
```

You can then look up the values for ‘name’ and ‘craft’

You should see something like:

```
People in Space:
Yuri Malenchenko
Timothy Kopra
Timothy Peake
```

```
print('People in Space: ', result['number'])

people = result['people']

for p in people:
  print(p['name'])
```

You are using live data so your results will depend on the number of people currently in space.

Challenge: Show the Craft

As well as the name of the astronaut the web service also provides the craft that they are in (such as the ISS.)

Can you add to your script so that it also prints out the craft that the astronaut is in.

Example:

```
People in Space: 3
Yuri Malenchenko in ISS
Timothy Kopra in ISS
Tim Peake in ISS
```

Step 2: Where is the ISS?

The International Space Station is in orbit around Earth. It orbits the earth roughly every hour and a half. The ISS travels at an average speed of 7.66 km per second. Let's use another web service to find out where the International Space Station is.

1 First open the url for the web service in a new tab in your web browser:

http://api.open-notify.org/iss-now.json

You should see something like this:

{"timestamp": 1569932551, "iss_position": {"longitude": "78.5006", "latitude": "51.2045"}, "message": "success"}

2 Refresh the page a few times to see the position constantly being updated. The result contains the coordinates of the spot on Earth that the ISS is currently over. Longitude is the East-West position and runs from -180 to 180. 0 is the Prime Meridian which runs through Greenwich in London, UK.

3 Latitude is the North-South position and runs from 90 to -90. 0 is the Equator. Now you need to call the same web service from Python. Add the following code to the end of your script to get the current location of the ISS and make variables to store and display its positions.

```
url = 'http://api.open-notify.org/iss-now.json'
response = urllib.request.urlopen(url)
result = json.loads(response.read())

location = result['iss_position']
lat = location['latitude']
lon = location['longitude']
print('Latitude: ', lat)
print('Longitude: ', lon)
```

```
Latitude:  26.4169023793
Longitude:  58.378453289
```

4 It would be more useful to show the position on a map. First we'll need to import the turtle graphics library.

```
import json
import urllib.request
import turtle
```

Let's load a world map as the background image, there's one already included in your trinket.

NASA have provided this map and given permission for reuse. The map is centered at 0, 0 which is just what you need. You need to **set the screen size** to match the size of the image which is **720 by 360.**

```
lon = location['longitude']
print('Latitude: ', lat)
print('Longitude: ', lon)

screen = turtle.Screen()
screen.bgpic('map.jpg')
```

5 Add "**screen.setup(720, 360)"** and **"screen.setworldcoordinates (-180,-90, 180, 90)"**

```
screen = turtle.Screen()
screen.setup(720, 360)
screen.setworldcoordinates(-180, -90, 180, 90)
screen.bgpic('map.jpg')
```

You want to be able to send the turtle to a particular latitude and longitude. **Insert the two lines of code above**.

6 Let's **create a turtle** for the ISS. Your project includes 'iss.png' and 'iss2.png', try them both and see which one you prefer.

```
screen = turtle.Screen()
screen.setup(720, 360)
screen.setworldcoordinates(-180, -90, 180, 90)
screen.bgpic('map.jpg')

screen.register_shape('iss.png')
iss = turtle.Turtle()
iss.shape('iss.png')
iss.setheading(90)
```

7 The ISS starts off in the centre of the map, now let's move it to the correct location on the map.

```
screen.register_shape('iss.png')
iss = turtle.Turtle()
iss.shape('iss.png')
iss.setheading(90)

iss.penup()
iss.goto(lon, lat)
```

Note that latitude is normally given first, but we need to give longitude first when plotting (x,y) coordinates.

8 **Test your program** by running it. The ISS should move to its current location above Earth. Wait a few seconds and run your program again to see where the ISS has moved to.