Curiosidades: el módulo *turtle*

El módulo **turtle** que podemos incorporar a nuestros programas Python nos deja utilizar el lenguaje para dibujar figuras en una pantalla gráfica, al estilo de un antiguo lenguaje de programación llamado Logo.

1. Primeros pasos con *turtle*

Lo primero que debemos hacer para poder utilizar este módulo es importarlo (`import`) en un programa Python:

```
import turtle
```

> **NOTA**: es importante que el archivo fuente que creemos no se llame *turtle*, ya que de lo contrario Python lo confunde con la librería original y no importa bien esta última.

A partir de aquí, debemos crear una variable de tipo *tortuga*, invocando a la instrucción `Turtle()` del módulo que hemos importado:

```
tortuga = turtle.Turtle()
```

2. Instrucciones básicas de manejo

Una vez tenemos disponible nuestra variable *tortuga*, podemos invocar una serie de instrucciones básicas desde esa variable:

- `forward(distancia)` / `fd(distancia)` : hace que la tortuga avance la distancia indicada
- `backward(distancia)` / `bk(distancia)` : hace que la tortuga retroceda la distancia indicada
- `right(angulo)` / `rt(angulo)` : hace que la tortuga gire a la derecha el ángulo indicado (en grados)
- `left(angulo)` / `lt(angulo)` : hace que la tortuga gire a la izquierda el ángulo indicado (en grados)
- `penup()` / `pu()` / `up()` : hace que la tortuga se "levante" de la pantalla, con lo que no dejará trazo al moverse
- `pendown()` / `pd()` / `down()` : hace que la tortuga baje, dejando trazo al moverse
- `color(nombre)` : cambia el color de trazo al indicado
- `fillcolor(nombre)` : cambia el color de relleno al indicado
- `clear()` : limpia el lienzo y deja la tortuga en su última ubicación

Por ejemplo, este fragmento dibuja un cuadrado de lado 50 en la pantalla:

```
import turtle
tortuga = turtle.Turtle()

for i in range(4):
    tortuga.forward(50)
    tortuga.right(90)

turtle.done()
```

> **NOTA**: la invocación de la instrucción `done()` del módulo *turtle* se produce al finalizar el programa.

2.1. Sobre los colores y rellenos

Algunas instrucciones, como `color` o `fillcolor` necesitan que especifiquemos un color (de trazo o relleno, respectivamente). Este color se puede especificar como un texto en inglés, como el color en hexadecimal, o indicando las componentes RGB del color, con valores de 0 a 1 separados por comas y entre paréntesis. Por ejemplo:

```
tortuga.color("red")
...
tortuga.color("#32c18f")
...
tortuga.fillcolor((0.2, 0.8, 0.7))
```

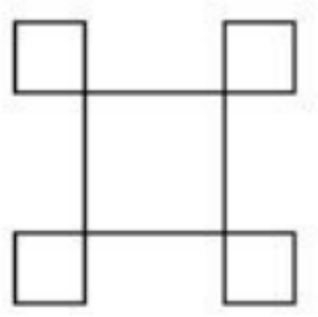
En el caso del color de relleno, debemos usar las instrucciones `begin_fill()` y `end_fill()` antes y después de haber dibujado la figura que queramos rellenar. Junto con la instrucción `fillcolor()`, que establece el color de relleno.

El siguiente ejemplo dibuja un rectángulo y lo rellena de rojo:

```
import turtle
tortuga = turtle.Turtle()

tortuga.begin_fill()

for i in range(4):
    tortuga.forward(50)
    tortuga.right(90)

tortuga.fillcolor("red")
tortuga.end_fill()

turtle.done()
```

Ejercicio 1:

Crea un archivo llamado `cuadrados.py` y utiliza *turtle* para dibujar una figura como esta:



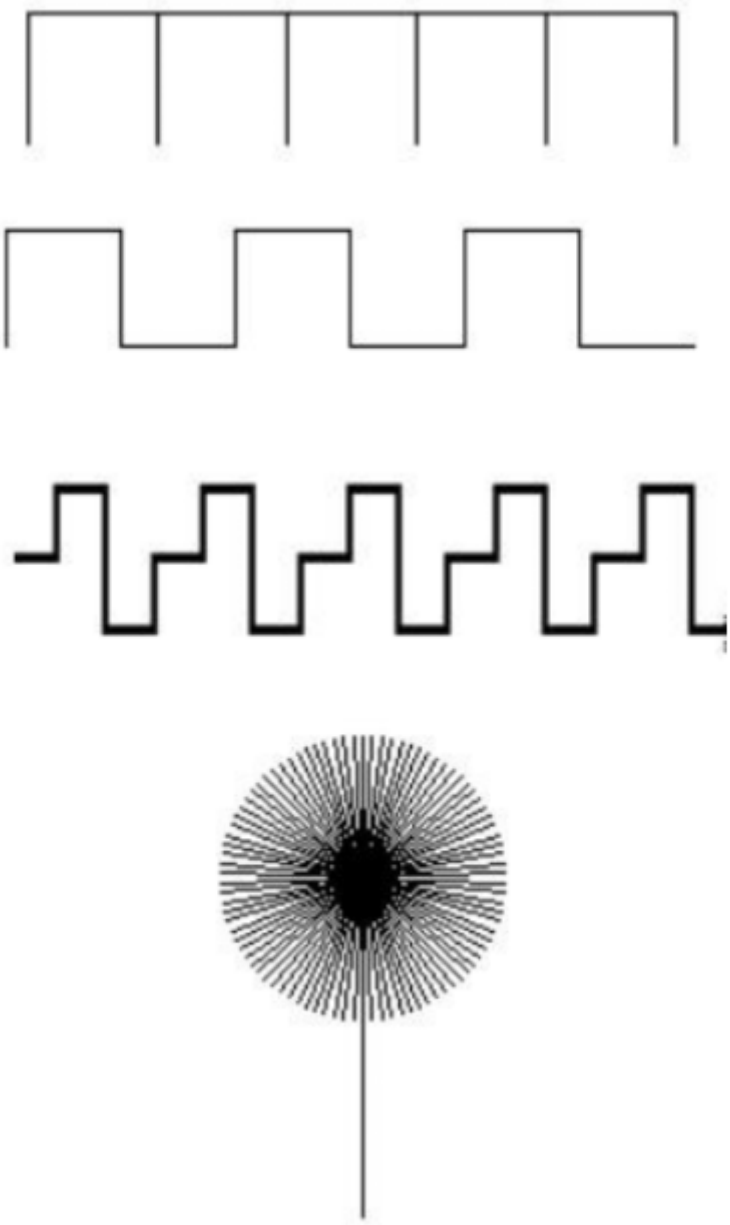
Ejercicio 2:

Crea un archivo llamado `pentagono.py` y utiliza *turtle* para dibujar un pentágono relleno de azul como este:

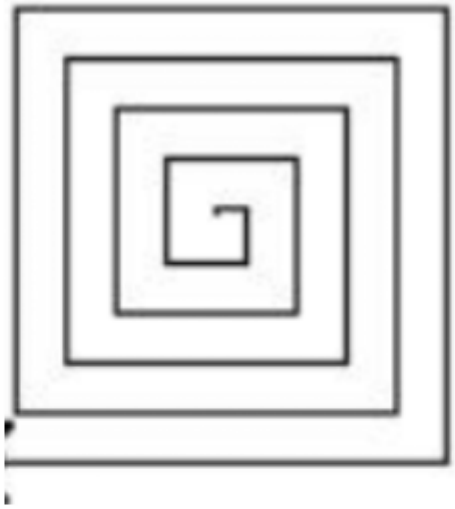
AYUDA: si sumamos todos los ángulos internos de un pentágono, la suma es de 540 grados.

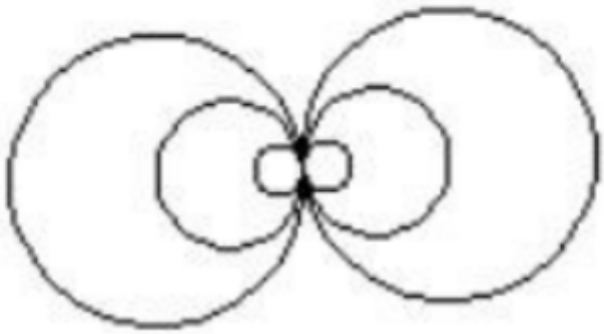
Ejercicio 3:

Dibuja las siguientes figuras en los archivos `peine.py`, `arriba_abajo.py`, `edificios.py` y `flor.py`, respectivamente



> **Ejercicio 4:**
>
> Dibuja las siguientes figuras en los archivos `espiral.py` y `circulos.py`:





> **AYUDA**: para dibujar un círculo, debes repetir varias veces avanzar un poco y girar un poco. Por ejemplo, si repites 120 veces avanzar 1 y girar 3, completarás un círculo. Si repites 360 veces avanzar 1 y girar 1, completarás un círculo mayor que el anterior.