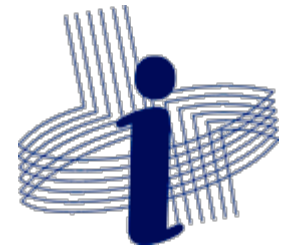


Universidade Federal de Viçosa
Departamento de Informática
Centro de Ciências Exatas e Tecnológicas

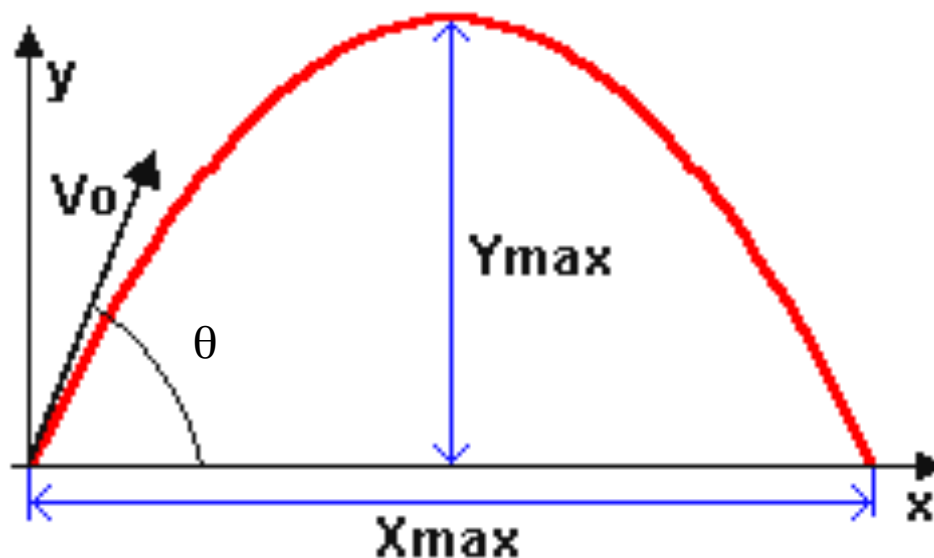


INF 100 – Introdução à Programação

Comandos de Repetição
- Desenhando na Tela -

Plotando gráficos simples

- Um projétil é lançado num terreno plano com velocidade inicial v_0 , segundo um ângulo θ em relação ao eixo horizontal (lançamento oblíquo), estando sob a ação da aceleração da gravidade g , agindo verticalmente para baixo, impondo uma trajetória parabólica, como mostra a figura abaixo.



Plotando gráficos simples

- Se ignorarmos a resistência do ar, podemos determinar o alcance máximo e a altura máxima do projétil usando as seguintes expressões:

$$x_{max} = \frac{v_0^2 \sin(2\theta)}{g} \qquad y_{max} = \frac{(v_0 \sin\theta)^2}{2g}$$

- Para calcularmos a altura do projétil em função de sua posição no eixo x , podemos usar a seguinte expressão:

$$y = x \tan \theta - 0,5g \left(\frac{x}{v_0 \cos \theta} \right)^2$$



Plotando gráficos simples

- Leia v_0 (em m/s), θ (em graus)
- Converta θ para radianos
- Calcule a altura máxima y_{max} e o alcance máximo x_{max} alcançados pelo projétil (para definir a escala do gráfico)
- Plote a altura do projétil em função da distância x , com x variando de 0 até x_{max} .



Plotando gráficos simples

- Plote a altura do projétil em função da distância x , com x variando de 0 até x_{max} .
Dividindo a escala por 100, temos:

```
delta_x ← x_max / 100
```

```
x ← 0
```

```
Enquanto x ≤ x_max:
```

```
    y ← x * tg( teta ) - 0.5g*(x / (v0*cos( teta )))2
```

```
    plotar( x, y )
```

```
    x ← x + delta_x
```

```
fim_enquanto
```



```
g = 9.80665
```

```
# leitura dos dados
```

```
v0 = float( input('Velocidade inicial (m/s) = '))
```

```
teta = float( input('Ângulo Teta (graus) = '))
```

```
# converter teta para radianos
```

```
teta = math.radians( teta )
```

```
# Para definir a escala do gráfico:
```

```
# calcular a altura máxima alcançada pelo projétil (teta = 90°)
```

```
y_max = (v0 * math.sin( math.pi / 2 ))**2 / (2*g)
```

```
# calcular a distância máxima alcançada pelo projétil (teta = 45°)
```

```
x_max = v0**2 * math.sin( 2*(math.pi / 4) ) / g
```

```
# Parâmetros dos eixos de plotagem
```

```
init_plot('Y x X', 'Distância (m)', 'Altura (m)',\  
          0, x_max, 0, y_max )
```



Plotando gráficos simples

calcular o alcance máximo do projétil

```
x_max = v0**2 * math.sin( 2*teta ) / g
```

plotar y em função de x

```
delta_x = x_max / 100
```

```
x = 0
```

```
while x <= x_max:
```

```
    y = x * math.tan( teta ) - 0.5 * g *  
        (x / (v0*math.cos( teta )))**2
```

```
    plot( x, y )
```

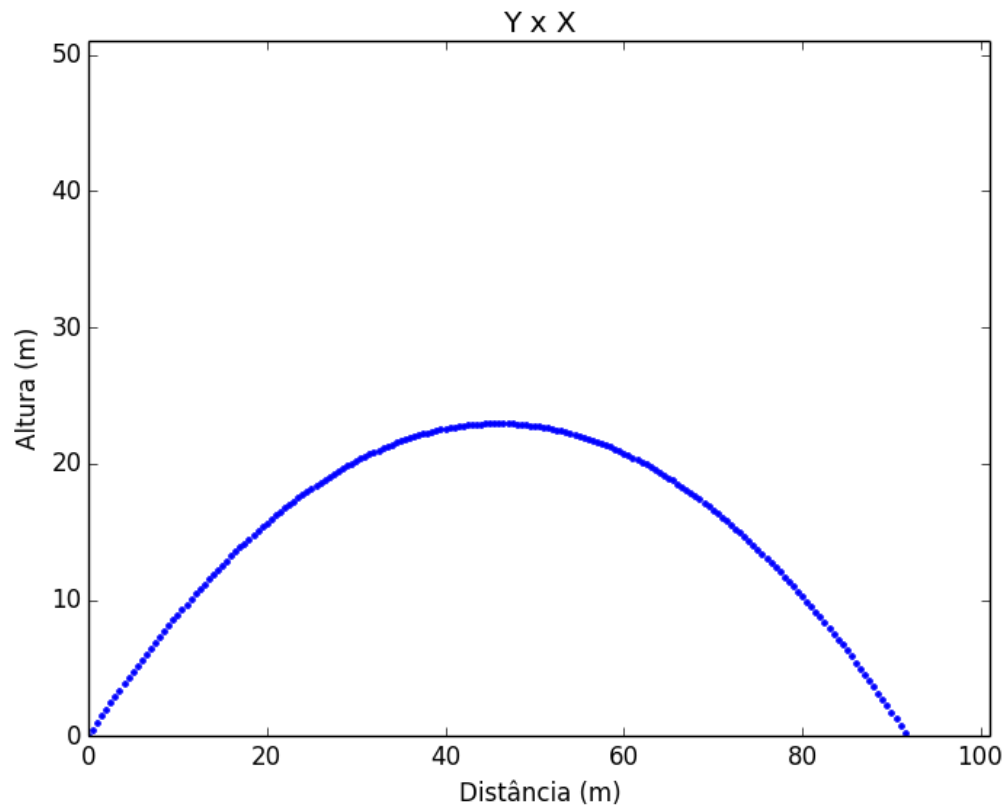
```
    x = x + delta_x
```



Plotando gráficos simples

Velocidade inicial (m/s) = 30

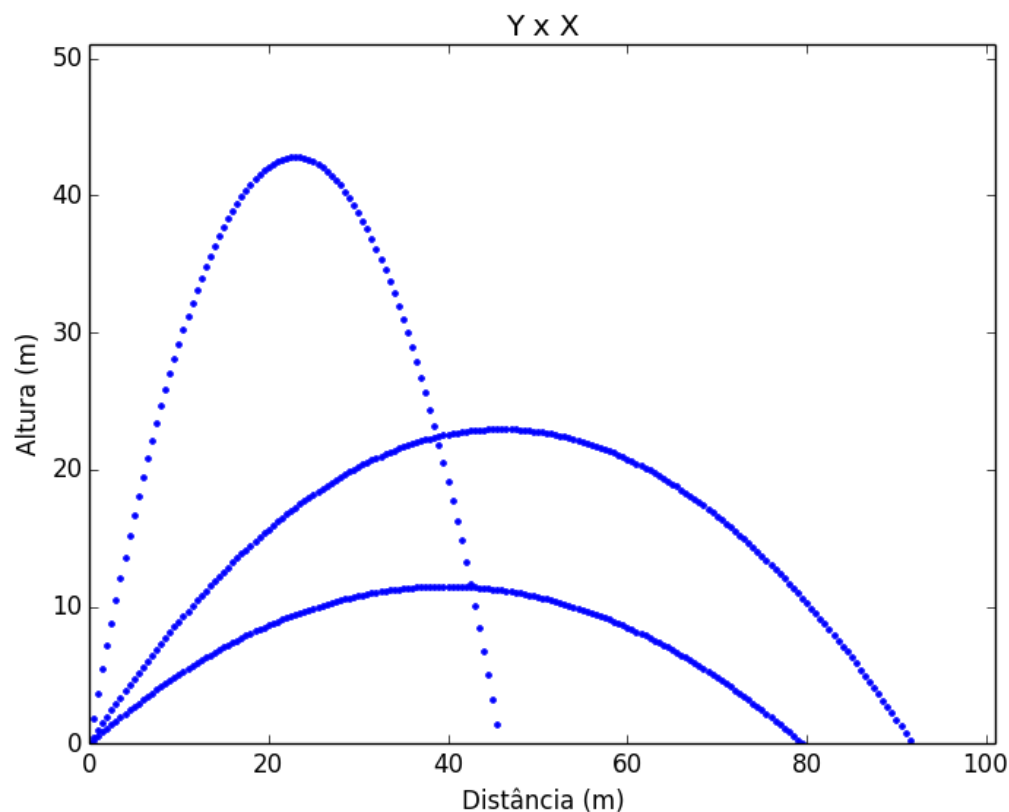
Ângulo Teta (graus) = 45



Plotando gráficos simples

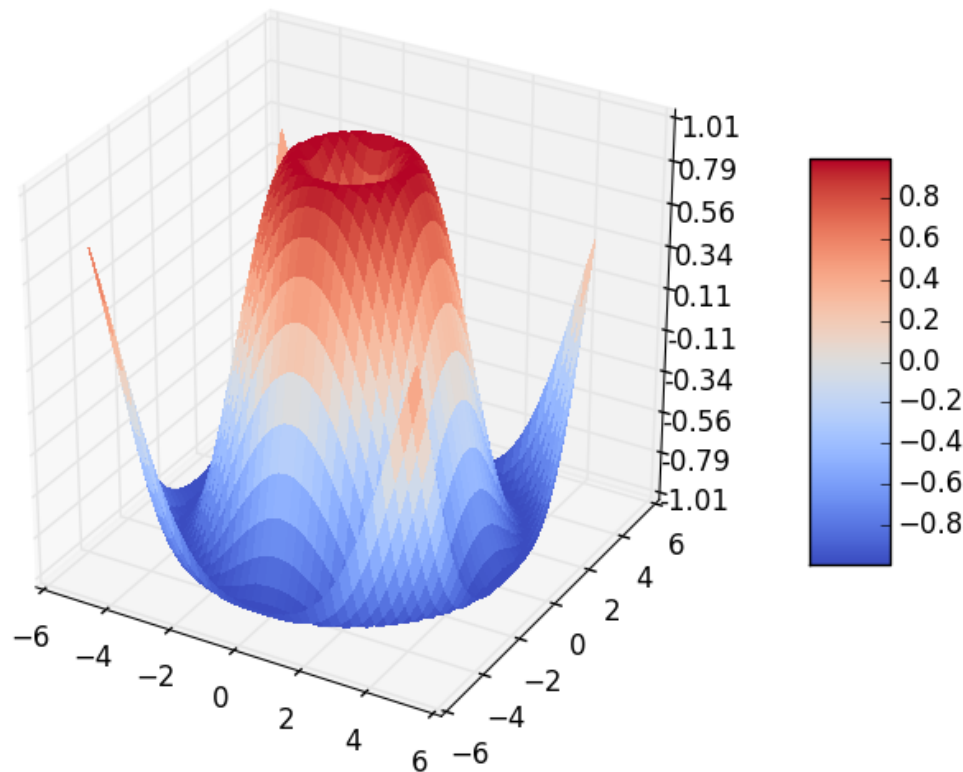
Velocidade inicial (m/s) = 30

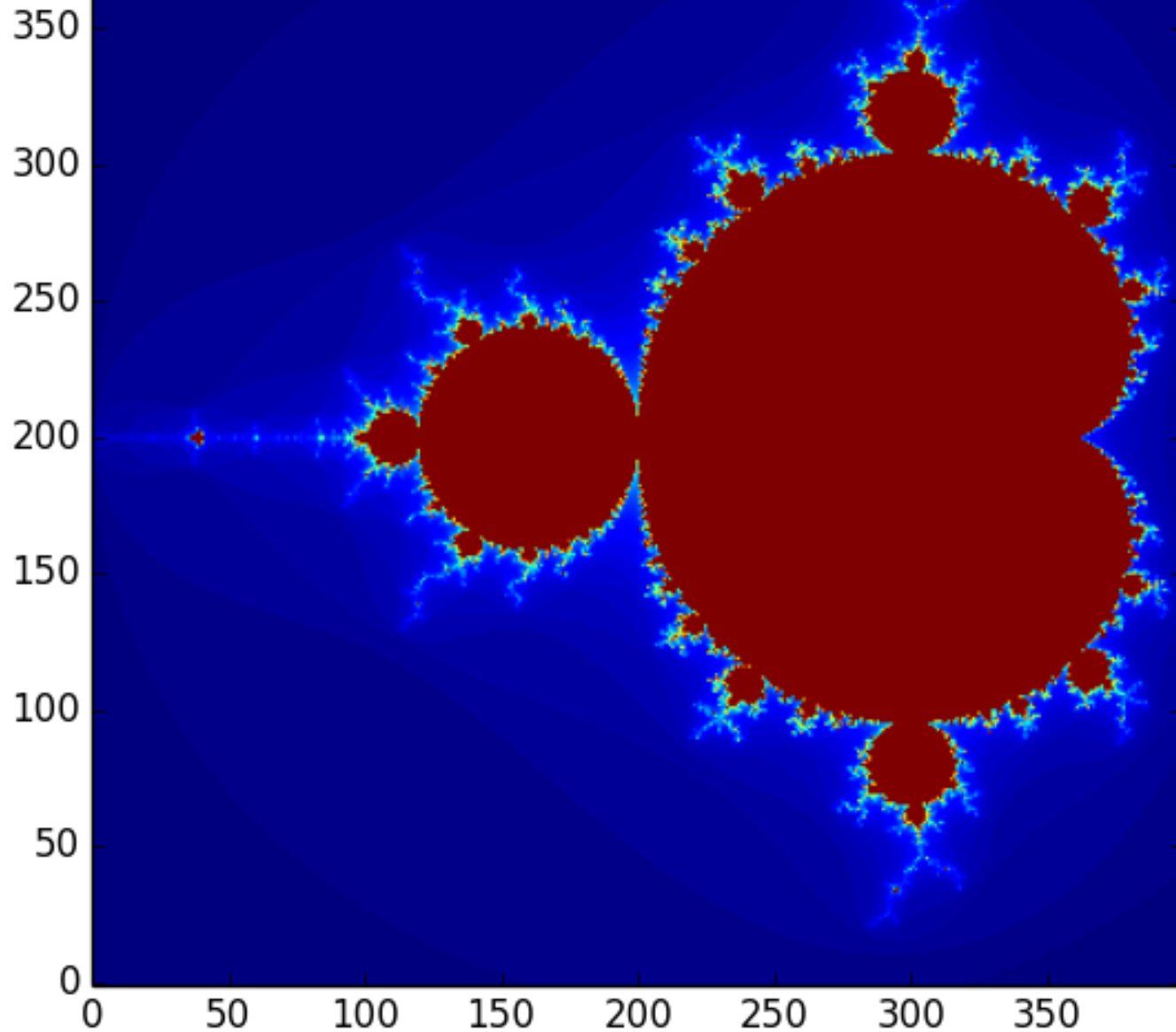
Ângulo Teta (graus) = 45, 30 e 75



Plotando gráficos 3D

- Com o PyPlot é possível plotar gráficos 3D, fazer simulações animadas etc.




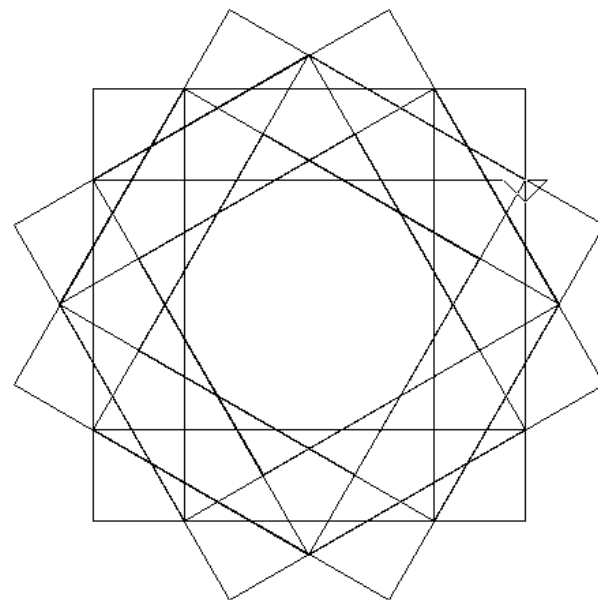


Fractal de
Mandelbrot



Mexendo a Tartaruga (*turtle graphics*)

- Linguagem de programação Logo
 - Criada em 1967 por Daniel G. Bobrow, Wally Feurzeig, Seymour Papert e Cynthia Solomon como ambiente de aprendizagem construtivista.
 - Tartaruga gráfica 
(funciona como um cursor ou caneta que se movimenta pela tela, desenhando figuras)



Mexendo a Tartaruga (*turtle graphics*)

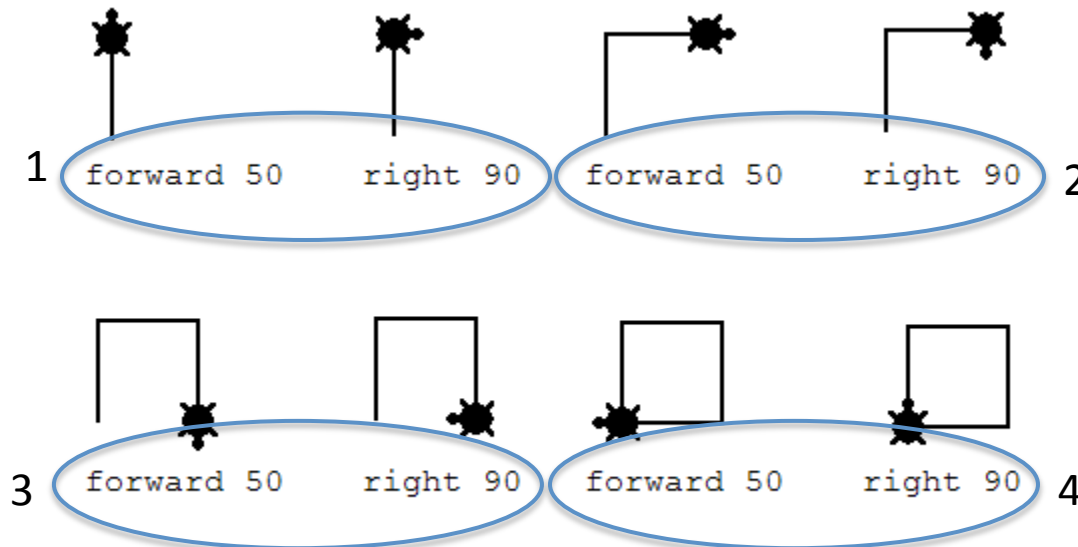
- Linguagem orientada a comandos gráficos

Principais Comandos (exemplos)	Efeito
forward 50	mover a caneta 50 pixels para frente
back 50	mover a caneta 50 pixels para trás
left 90	girar o cursor 90° para a esquerda
right 90	girar o cursor 90° para a direita
up	tirar a caneta do 'papel'
down	encostar a caneta no 'papel'
color ou pencolor	mudar a cor da caneta



Mexendo a Tartaruga (*turtle graphics*)

- Exemplo:



Mexendo a Tartaruga (*turtle graphics*)

- Para desenhar um quadrado de lado L:
 - Repetir 4 vezes:
 - `forward L`
 - `right 90`



Tartarugas em Python

- Usando a biblioteca 'turtle', podemos usar gráficos de tartaruga dentro de Python.

```
import turtle
```



Alguns comandos e funções para a Tartaruga em Python

Comando	Efeito
<code>forward(n), fd(n)</code>	andar n passos para frente
<code>backward(n), back(n), bk(n)</code>	andar n passos para trás
<code>left(a), lt(a)</code>	virar a graus para a esquerda
<code>right(a), rt(a)</code>	virar a graus para a direita
<code>pos()</code>	Retorna as coordenadas x, y atuais da tartaruga
<code>goto(x, y)</code>	Mover a tartaruga para as coordenadas x, y .
<code>home()</code>	Mover a tartaruga para o centro da tela (posição inicial)
<code>speed(n)</code>	Ajustar a velocidade da tartaruga
<code>pendown(), pd(), down()</code>	Abaixar a caneta
<code>penup(), pu(), up()</code>	Suspender a caneta
<code>pensize(n), width(n)</code>	Ajustar a espessura da caneta
<code>pencolor(cor)</code>	Ajustar a cor da caneta



Tartarugas em Python

- Manual de Referência completo:

<https://docs.python.org/release/3.1.3/library/turtle.html>



Tartarugas em Python

- Desenhando um quadrado:

```
import turtle as t
```

```
L = int(input('Tamanho do quadrado: '))
```

```
for i in range( 0, 4 ):
    t.forward( L )
    t.right( 90 )
```

```
t.Screen().exitonclick()
```



Tartarugas em Python

- Desenhando um polígono regular qualquer:

```
import turtle as t

n = int(input('Entre com o numero de lados do polígono: '))
ang_interno = (n-2)*180/n # Ângulo interno do polígono

for i in range( 0, n ):
    t.forward( 100 )
    t.right( 180-ang_interno )

t.Screen().exitonclick()
```



Desenhar quadrados concêntricos

```
import turtle as t
```

```
h = int(input('Tamanho do quadrado externo: '))
```

```
t.speed( 5 )
```

```
t.pencolor('blue')
```

```
while h > 0:
```

```
    for i in range( 0, 4 ):
```

```
        t.forward( h )
```

```
        t.right( 90 )
```

```
    t.up()
```

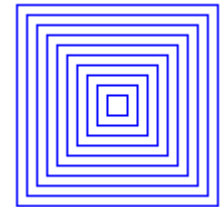
```
    t.goto( t.xcor()+5, t.ycor()-5 )
```

```
    t.down()
```

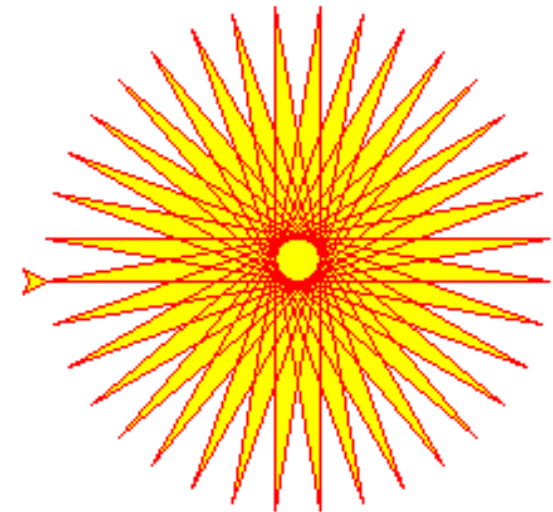
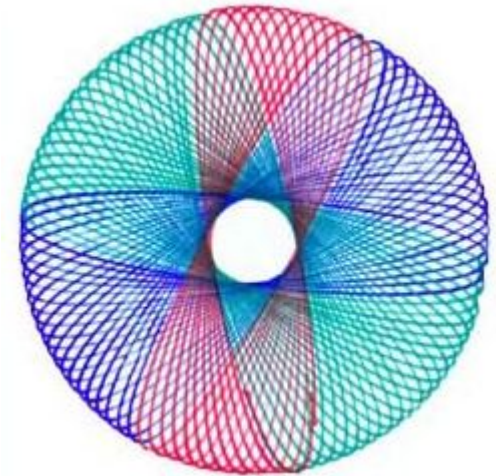
```
    h = h - 10
```

```
t.hideturtle()
```

```
t.Screen().exitonclick()
```



Desenhar uma 'flor' (já brincaram com o 'spirograph'?)



'Spirograph set (UK Palitoy early 1980s) (perspective fixed)' by Multicherry. Licensed under CC BY-SA 3.0 via Wikimedia Commons

Tartarugas em Python

```
# Desenhar "flor" amarela/vermelha

import turtle as t

t.speed( 5 ) # 1-10, sendo 0 = sem "lag"
t.color('red', 'yellow')
t.begin_fill()

while True:
    t.forward(200)
    t.left(170)
    if abs(t.pos()) < 1:
        break

t.end_fill()

t.Screen().exitonclick()
```

