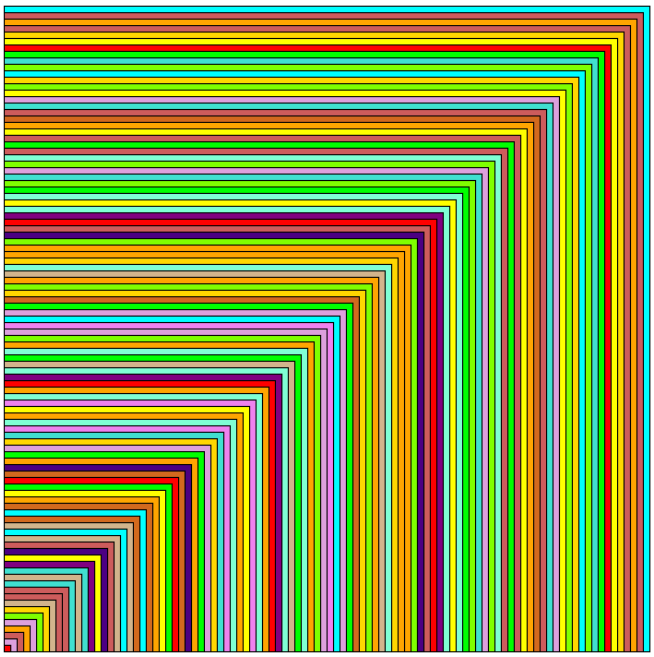
Exercices sur Turtle avec remplissage aléatoire de couleurs

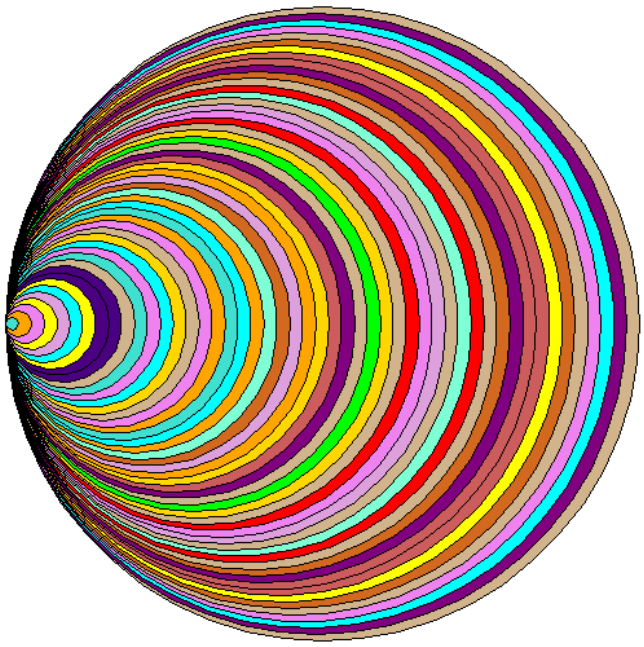
Carres emboités

```
from turtle import *
from random import *
speed(40)
colors = ["yellow","cyan","purple","turquoise",
"orange","red","lime","indianred","gold",
"chartreuse","aquamarine","indigo","violet",
"chocolate","tan","plum"]

def dessiner_un_carre(l):
    for i in range(4):
        forward(l)
        left(90)

def dessiner_carres_emboites(n):
    for i in range(n):
        fillcolor(choice(colors))
        begin_fill()
        dessiner_un_carre(600 - 6*i)
        end_fill()

def placer_le_curseur(x,y):
    up()
    setposition(x, y)
    down()

placer_le_curseur(-300,-300)
dessiner_carres_emboites(100)
hideturtle()
```



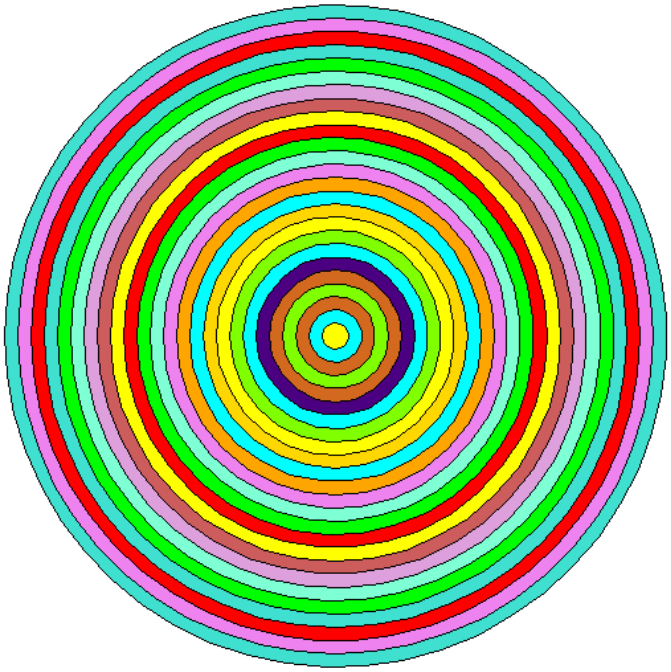
Cercles emboités

```
from turtle import *
from random import *
speed(15)
up()
back(450)
right(90)
down()
colors = ["yellow","cyan","purple","turquoise",
"orange","red","lime","indianred","gold",
"chartreuse","aquamarine","indigo","violet",
"chocolate","tan","plum"]
rayon = 5
nb = 50
for i in range(1, nb + 1):
    fillcolor(choice(colors))
    begin_fill()
    circle(250 - rayon*i)
    end_fill()
hideturtle()
```



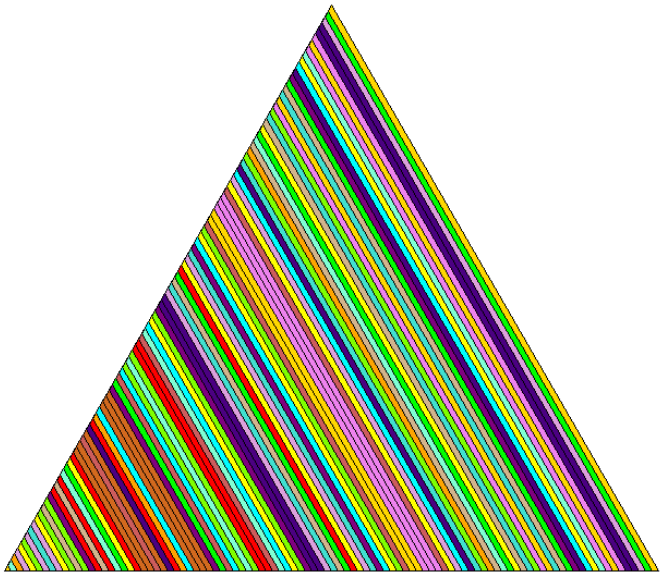
Cercles concentriques

```
from turtle import *
from random import *
speed(50)
up()
back(450)
right(90)
down()
colors = ["yellow","cyan","purple","turquoise",
"orange","red","lime","indianred","gold",
"chartreuse","aquamarine","indigo","violet",
"chocolate","tan","plum"]
rayon = 10
nb = 25
for i in range(0, nb):
    fillcolor(choice(colors))
    begin_fill()
    circle(250 - rayon*i)
    end_fill()
    up()
    left(90)
    forward(rayon)
    right(90)
    down()
hideturtle()
```



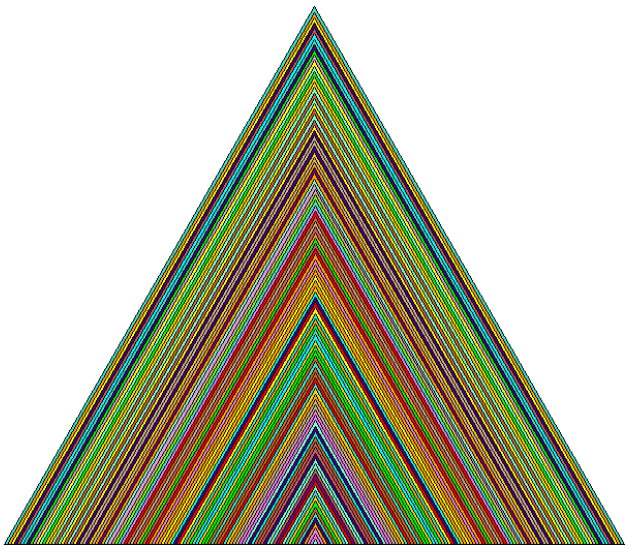
Triangles emboités

```
from turtle import *
from random import *
speed(40)
colors = ["yellow","cyan","purple","turquoise",
"orange","red","lime","indianred","gold",
"chartreuse","aquamarine","indigo","violet",
"chocolate","tan","plum"]

def dessiner_un_triangle(l):
    for i in range(3):
        forward(l)
        left(120)

def dessiner_triangles_emboites(n):
    for i in range(n):
        fillcolor(choice(colors))
        begin_fill()
        dessiner_un_triangle(600 - 6*i)
        end_fill()

def placer_le_curseur(x,y):
    up()
    setposition(x, y)
    down()

placer_le_curseur(-300,-300)
dessiner_triangles_emboites(100)
hideturtle()
```



Triangles emboités centrés

```
from turtle import *
from random import *
speed(40)
colors = ["yellow","cyan","purple","turquoise",
"orange","red","lime","indianred","gold",
"chartreuse","aquamarine","indigo","violet",
"chocolate","tan","plum"]

def dessiner_un_triangle(l):
    for i in range(3):
        forward(l)
        left(120)

def dessiner_triangles_emboites(n):
    for i in range(n):
        fillcolor(choice(colors))
        begin_fill()
        dessiner_un_triangle(600 - 6*i)
        end_fill()
        up()
        forward(600/199)
        down()

def placer_le_curseur(x,y):
    up()
    setposition(x, y)
    down()

placer_le_curseur(-300,-300)
dessiner_triangles_emboites(100)
hideturtle()
```

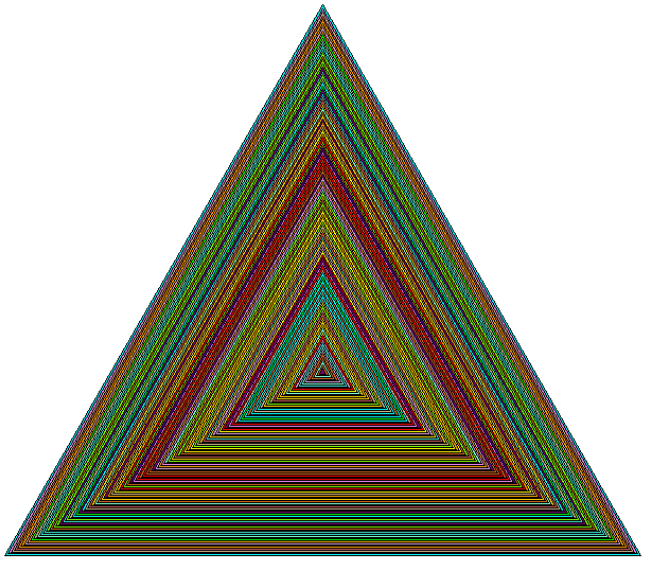


Explications :
Nous traçons 100 triangles imbriqués,
le plus petit est de largeur 1
le deuxième de largeur 3,
...
le n-ième de largeur $2n-1$,
le $100^{ème}$ *de largeur 199.*
Or le plus grand est de largeur réelle 600
Il faut donc avancer d'une valeur égale à :

$$\frac{600}{199}$$

Triangles emboités centrés de gravité

```
from turtle import *
from random import *
from math import *
speed(40)
colors = ["yellow","cyan","purple","turquoise",
"orange","red","lime","indianred","gold",
"chartreuse","aquamarine","indigo","violet",
"chocolate","tan","plum"]
def dessiner_un_triangle(l):
    for i in range(3):
        forward(l)
        left(120)
def dessiner_triangles_emboites(n):
    for i in range(n):
        fillcolor(choice(colors))
        begin_fill()
        dessiner_un_triangle(600 - 6*i)
        end_fill()
        up()
        left(30)
        forward(sqrt(270000)*2/3/100)
        right(30)
```



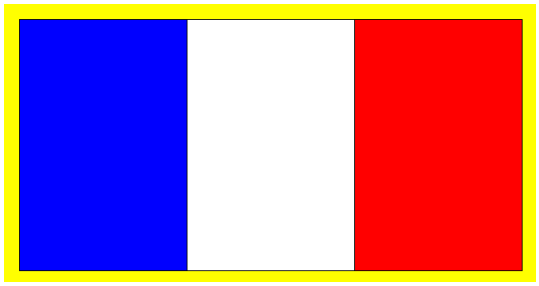
Explications :
Avec Pythagore, la hauteur/médiane mesure
$\sqrt{600^2-300^2}=\sqrt{270\ 000}$.
La distance d'un sommet jusqu'au centre de gravité est égale aux deux-tiers de la médiane

```
        down()
def placer_le_curseur(x,y):
    up()
    setposition(x, y)
    down()
placer_le_curseur(-300,-300)
dessiner_triangles_emboites(100)
hideturtle()
```

qui se trouve à 30°.
On divise l'avancement par le nombre de triangles

Drapeau français

```
from turtle import *
speed(10)
up()
back(300)
down()
def panneau() :
    forward(200)
    left(90)
    forward(300)
    left(90)
bgcolor("yellow")
fillcolor("blue")
begin_fill()
for i in range(2):
    panneau()
end_fill()
forward(200)
fillcolor("white")
begin_fill()
for i in range(2):
    panneau()
end_fill()
forward(200)
fillcolor("red")
begin_fill()
for i in range(2):
    panneau()
end_fill()
hideturtle()
```



Drapeau français

```
from PIL import Image
import PIL
img=Image.new(("RGB"),(900,450),"grey")
for x in range(300):
    for y in range(450):
        img.putpixel((x,y),(0,0,255))
for x in range(300,600):
    for y in range(450):
        img.putpixel((x,y),(255,255,255))
for x in range(600,900):
    for y in range(450):
        img.putpixel((x,y),(255,0,0))
img.show()
img.save("Drapeau_Français_programme_Python.png")
```