```
import turtle
turtle.tracer(0,0)
turtle.screensize(2000,2000)
turtle.pu()
turtle.goto(-500,0)
turtle.pd()


def dessiner(courbe, longueur, angle):
  """ réalise une représentation graphique d'une courbe donnée par des chaines de c
aractères """
  for caractere in courbe:
    if caractere == '+': turtle.left(angle)
    elif caractere == '-': turtle.right(angle)
    elif caractere in ['F', 'G']: turtle.forward(longueur)

#dessiner('F', 50, 60)


def regleSierpinski(chaine):
  nouvelleChaine = ''
      # on crée une nouvelle chaine de caractères VIDE
  for lettre in chaine:
      # on épelle la chaine de caractères donnée en paramètres
    if lettre == 'F':
      nouvelleChaine = nouvelleChaine + 'F-G+F+G-F'
    elif lettre in ['-' , '+'] :
      nouvelleChaine = nouvelleChaine + lettre
    else :
      nouvelleChaine = nouvelleChaine + 'GG'
  return nouvelleChaine


def courbeSierpinski(motifInitial, niter):
  """
    appelle niter fois regleSierpinski pour créer la courbe de Sierpinski
  """
  courbe = motifInitial
# on part du motif initial
  for i in range(niter):
    nouveauMotif = regleSierpinski(courbe)
#on trouve le nouveau Motif à partir de celui initial
    courbe = nouveauMotif
  return courbe
```

```
#courbe = courbeSierpinski('F',3)
#dessiner(courbe,50, 60)



def triangle(motifInitial, niter):
  courbe = courbeSierpinski(motifInitial, niter)
  triangle = ''
  for _ in range(3):
    triangle += courbe
    triangle += '-'
  return triangle



chaine = 'F-G-G'
longueur = 30
angle = 120
niter = 3


dessiner(courbeSierpinski('F-G-G', niter), longueur, angle)


turtle.update()
turtle.exitonclick()
```