This is an enhanced version of the chain (catenary curve) example discussed in Chapter 5 of the book Object-Oriented Programming in Python. Authors: Jason Fritts, Michael Goldwasser

```python
from cs1graphics import *
from math import sqrt

# Define constants

def combine(A, B, C=(0,0)):

def calcForce(A, B):
    dX = B[0] - A[0]
    distance = sqrt(dX * dX + dY * dY)
    if distance > restingLength:
        stretch = distance - restingLength
        forceFactor = stretch * elasticityConstant
    else:
        forceFactor = 0
    return (forceFactor * dX, forceFactor * dY)

def drawChain(chainData, chainPath, theCanvas):
    for k in range(len(chainData)):
        chainPath.setPoint(Point(chainData[k][0], chainData[k][1]), k)
    theCanvas.refresh()

# Initialize the chain; one end at (0,0) other at (totalSeparation,0)
chain = []
for k in range(numLinks + 1):
    X = totalSeparation * k / numLinks
    chain.append( (X, 0.0) )
    chain[chain[-1]].setPoint(Point(chain[-1][0], chain[-1][1]), k)
    chain[chain[-1]].refresh()
```

We will only draw some iterations.
# as long as forces are not (sufficiently) in equilibrium, adjust
# chain link positions and re-draw

somethingMoved = True  # force loop to start

while somethingMoved:
    somethingMoved = False  # default for new iteration
    oldChain = list(chain)  # record a copy of the data

# examine forces being applied to a chain link, and
# adjust link position if not at equilibrium

for k in range(1, numLinks):
    gravForce = (0, gravityConstant)  # downward force
    leftForce = calcForce(oldChain[k], oldChain[k-1])
    rightForce = calcForce(oldChain[k], oldChain[k+1])
    adjust = combine(gravForce, leftForce, rightForce)
    chain[k] = combine(oldChain[k], adjust)

    if abs(adjust[0]) > epsilon or abs(adjust[1]) > epsilon:
        somethingMoved = True

# only draw intermittent results, otherwise display takes too long
if graphicsCounter == 0:
    drawChain(chain, curve, paper)
    graphicsCounter = int(25 * speed)

# display final result with line emphasized (bold)
curve.setBorderWidth(2)
drawChain(chain, curve, paper)