```
import numpy as np
import math
from scipy import signal
import ncc
from PIL import Image, ImageDraw

scale = 0.75 # given scale factor
templateWidth = 15

"""
Returns a pyramid for an image and minsize provided, gives a list of the original image, then
reduced and scaled versions 3/4 the previous size.
"""
def MakePyramid(image, minsize):
    # init list where we will store the images and place the original image in it.
    pyramid = []
    pyramid.append(image)

    # check width and height dimensions are > minimum.
    while(image.size[0] > minsize and image.size[1] > minsize):
        x = image.size[0]
        y = image.size[1]
        # Make a new copy with scaled dimensions
        image = image.resize((int(x*scale),int(y*scale)),Image.BICUBIC)
        pyramid.append(image)

    return pyramid

"""
Displays the given pyramid.
"""
def ShowPyramid(pyramid):

    # canvas on which we display our image will have the height of the first (largest) image.
    height = pyramid[0].size[1]
    width = 0
    # iterate through all the images and keep count of the combined width of the images.
    for image in pyramid:
        width += image.size[0]

    # the image where we're gonna post everything
    canvas = Image.new("L", (width, height),"white")

    # copy images in the pyramid onto the canvas at right places
    offset = 0
    for image in pyramid:
        canvas.paste(image,(offset,0))
        # Increment the offset at every step.
        offset += image.size[0]
```

```
    canvas.show()

"""
Finds and marks all locations within the pyramid where the NCC is above the given threshold, draws
the red rectangle surrounded the matches. Returns the marked image.
"""
def FindTemplate(pyramid, template, threshhold):
    # maintain a variable so we know by how much to scale the coordinates back.
    scaleBack = 1

    # this marker will be the image where we draw the rectangles of matches we find.
    marker = pyramid[0]
    marker = marker.convert('RGB')

    # loop over every image in the pyramid to find matches.
    for image in pyramid:
        # Get the normalized cross correlation of the template with the image.
        crossXC = ncc.normxcorr2D(image, template)

        # We loop through this 2D returned array to check for values larger than our threshold.
        for y in range(len(crossXC)):
            for x in range(len(crossXC[y])):
                if crossXC[y][x] > threshhold:

                    # find scaled coordinate values
                    draw = ImageDraw.Draw(marker)
                    x1 = x*scaleBack + template.size[0]
                    x2 = x*scaleBack - template.size[0]
                    y1 = y*scaleBack - template.size[1]
                    y2 = y*scaleBack + template.size[1]

                    # draw a red rectangle around
                    draw.line((x1,y1,x2,y1),fill="red",width=2)
                    draw.line((x1,y2,x2,y2),fill="red",width=2)
                    draw.line((x1,y1,x1,y2),fill="red",width=2)
                    draw.line((x2,y1,x2,y2),fill="red",width=2)

                    del draw
        # scaleback is determined the inverse of 0.75, our original scale value - (0.75)^(-1) = 1.33333333
        scaleBack *= 1.33333

    return marker


im = Image.open("/home/i/i7f7/cs425/a3/faces/judybats.jpg")
im = im.convert('L')
```

```
# build our pyramid and display it
pyramid = MakePyramid(im,15)
ShowPyramid(pyramid)
#Get our template and resize it based on the templateWidth variable defined at the beginning.
template = Image.open("/home/i/i7f7/cs425/a3/faces/judybats.jpg");
templateHeight = template.size[1]*templateWidth/template.size[0]
template = template.resize((int(templateWidth),int(templateHeight)),Image.BICUBIC)

# Find the marked image and display it.
# 0.51 Was the optimum threshold found
finalImage = FindTemplate(pyramid,template,0.51)
finalImage.show()
```