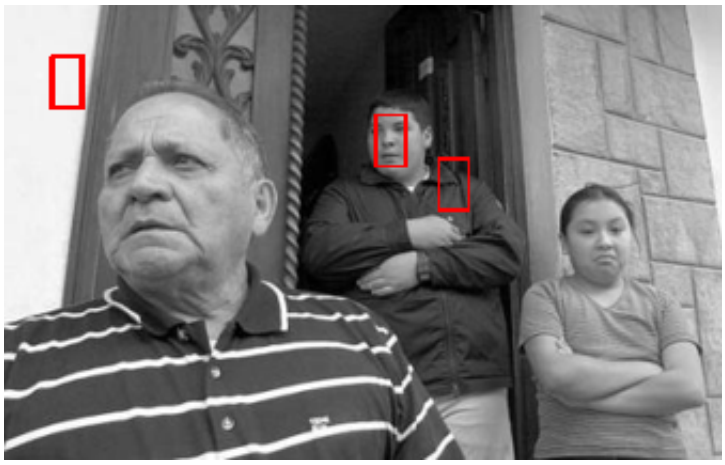
In [2]:
```
from PIL import Image, ImageDraw
import numpy as np
import math
from scipy import signal
from IPython.display import Image as Imag
import ncc
import os
```

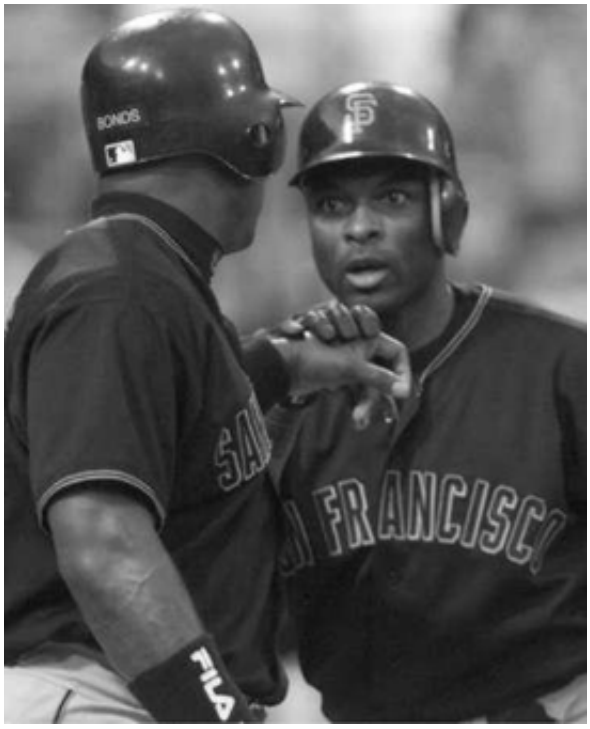
In [29]:
```
def MakePyramid(image,minsize):
#   Get the x and y sizes
    x,y = image.size
#   Store the reduced images in the list below
    images = []
#   Reduce original image by 3/4th
    while (x > minsize) and (y > minsize):
        new_img = image.resize((int(x*0.75), int(y*0.75)), Image.BICUBIC)
        images.append(new_img)
        x *= 0.75
        y *= 0.75
    return images
```

In [30]:
```
def ShowPyramid(pyramid):
#    dx and dy are offsets
     dx = 0
     dy = 0
#    get maximum height and sum of width in order to create the new image
     max_height = max([image.size[1] for image in pyramid])
     max_width = sum([image.size[0] for image in pyramid])
     new_image = Image.new("L", (max_width, max_height))
     for im in pyramid:
        new_image.paste(im,(dx,dy))
        dx += im.size[0]
     new_image.save('output.png')
     display(Imag(filename='output.png'))
image = Image.open("./faces/fans.jpg")
ShowPyramid(MakePyramid(image,15))
```

In [95]:
```
def FindTemplate(pyramid,template,threshold):
#   minimum possible template size.reduced template width as NCC is expen
    template_w = 15
#   resize template to
    scale = template.size[0]/template_w
    template = template.resize((template_w, template.size[1]//scale), Ima
    x,y = template.size
#   image must support color
    img = pyramid[0].convert('RGB')
#   loop over images in pyramid and get
    for im in pyramid:
#       get all the pixels that are above threshold
        thresh = np.where(ncc.normxcorr2D(im, template) > threshold)
#       For each pixel at which the normalized correlation result
#       is above the threshold, draw rectangle
        for i in range(len(thresh[1])):
                template_x = x/2
                template_y = y/2
                x1=thresh[1][i]-template_x
                y1=thresh[0][i]-template_y
                x2=thresh[1][i]+template_x
                y2=thresh[0][i]+template_y
                draw=ImageDraw.Draw(img)
                draw.rectangle((x1,y1,x2,y2),outline="red")
    img.save('result.png','PNG')
    display(Imag(filename='result.png'))
#     img.show()
    return img
```

In [98]:
```
allimages = ['judybats.jpg','students.jpg','tree.jpg','family.jpg','fans.
# change directory to faces
os.chdir("./faces/")
# open template
template = Image.open("./template.jpg")
# best threshhold
threshold = 0.58412
# For each image find the template
for image in allimages:
    image = str(os.getcwd()) + "/" +str(image)
    image = Image.open(image)
    pyramid = MakePyramid(image, 5)
    # Get template matches
    result = FindTemplate(pyramid, template, threshold)
os.chdir("..")
```




BONDS
FILA

```
A.5 Final threshold for equal error rate :threshold = 0.58412
An equal error rate is where the number of non-faces seen as faces
(false positives) equals the number of missed faces (false negatives).
```

```
A.6 Recall on the following images:
judybats = 3/5
students = 16/27
tree = undefined
family = 1/3
fans = 0/3
sports = 0/1

The NCC has a very low recall rate on some images due to different
scales,
different orientation,lighting conditions or maybe different
perspective.
```