The Project

1. This is a project with minimal scaffolding. Expect to use the the discussion forums to gain insights! It's not cheating to ask others for opinions or perspectives!
2. Be inquisitive, try out new things.
3. Use the previous modules for insights into how to complete the functions! You'll have to combine Pillow, OpenCV, and Pytesseract
4. There are hints provided in Coursera, feel free to explore the hints if needed. Each hint provide progressively more details on how to solve the issue. This project is intended to be comprehensive and difficult if you do it without the hints.

The Assignment

Take a ZIP file (https://en.wikipedia.org/wiki/Zip_(file_format)) of images and process them, using a library built into python (https://docs.python.org/3/library/zipfile.html) that you need to learn how to use. A ZIP file takes several different files and compresses them, thus saving space, into one single file. The files in the ZIP file we provide are newspaper images (like you saw in week 3). Your task is to write python code which allows one to search through the images looking for the occurrences of keywords and faces. E.g. if you search for "pizza" it will return a contact sheet of all of the faces which were located on the newspaper page which mentions "pizza". This will test your ability to learn a new (library (https://docs.python.org/3/library/zipfile.html)), your ability to use OpenCV to detect faces, your ability to use tesseract to do optical character recognition, and your ability to use PIL to composite images together into contact sheets.

Each page of the newspapers is saved as a single PNG image in a file called images.zip (./readonly/images.zip). These newspapers are in english, and contain a variety of stories, advertisements and images. Note: This file is fairly large (~200 MB) and may take some time to work with, I would encourage you to use small_img.zip (./readonly/small_img.zip) for testing.

Here's an example of the output expected. Using the small_img.zip (./readonly/small_img.zip) file, if I search for the string "Christopher" I should see the following image:

Christopher Search
If I were to use the images.zip (./readonly/images.zip) file and search for "Mark" I should see the following image (note that there are times when there are no faces on a page, but a word is found!):

Mark Search

Note: That big file can take some time to process - for me it took nearly ten minutes! Use the small one for testing.

In [1]:
```
import zipfile
import PIL
from PIL import Image
import pytesseract
import cv2 as cv
import numpy as np
from PIL import ImageDraw

# loading the face detection classifier
face_cascade = cv.CascadeClassifier('readonly/haarcascade_frontalface_default.xml')

# the rest is up to you!
```

In [2]:
```
#Step 1: Creating the ZipList
pages = zipfile.ZipFile("readonly/small_img.zip")

pageslist = pages.infolist()

for i in pageslist:
    print (i.filename)
```

```
a-0.png
a-1.png
a-2.png
a-3.png
```

In [3]:
```
#Step 2: Transforming each image into a pillow object and appending them to a list
pillimg = []

for i in pageslist:
    img = Image.open(pages.open(i))
    pillimg.append(img)
```

In [4]:
```
#Step 3: Creating the word recognition. Note: 10 minutes
textlist = []
for i in range(len(pillimg)):
    intcopy = pillimg[i]
    intcopy2 = intcopy.copy()
    intcopy2 = intcopy2.convert("1")
    text = pytesseract.image_to_string(intcopy2)
    textlist.append(text)

for i in textlist:
    print(i[0:200])
```

Ann Arbor, Michigan

Wednesday, November 5, 2014

michigandally.com

SNYDER EARNS SECOND TERM; G.O.P. TAKES CONTROL OF U.S. SENATE

2A — Wednesday, November 5, 2014

Students vote, watch Midterm Election 2014

The Michigan Daily ~— michigandaily.com

Che Michigan Daily

420 Maynard St.
Ann Arbor, MI 48109-1327
www.michiga
The Michigan Daily — michigandaily.com

Page 3A — Wednesday, November 5, 2014

an Daily

Edited and managed by students at
the U

4A, 5A — Wednesday, November 5, 2014

The Michigan Daily — michigandaily.com

LUNA ANNA ARCHEY/Daily

Ann Arbor Mayor elect Chris Taylor interacts with supporters at a watch party at

In [5]:
```
#Step 4: Creating the Face Recognition  #2 minutes
facecoor = []

for i in range(len(pillimg)):
    intcopy = pillimg[i]
    intcopy2 = intcopy.copy()
    open_cv_image = np.array(intcopy2)
    gray = cv.cvtColor(open_cv_image, cv.COLOR_BGR2GRAY)
    faces = face_cascade.detectMultiScale(gray, scaleFactor =1.3, minNeighbors = 5)
    faces = faces.tolist()
    faces2 = []
    for i in faces:
        x1, y1, x2, y2 = i[0], i[1], i[0] + i[2], i[1] + i[3]
        coor = (x1,y1,x2,y2)
        faces2.append(coor)
    facecoor.append(faces2)



print(facecoor)
```

```
[[(3139, 1733, 3419, 2013), (2545, 1957, 2743, 2155), (1150, 2000, 1363, 2213),
(1674, 2025, 1871, 2222), (1966, 1881, 2183, 2098), (2661, 3065, 2937, 3341)],
[(833, 2382, 1045, 2594), (2235, 2448, 2286, 2499), (2063, 2499, 2114, 2550), (2
515, 2419, 2576, 2480), (492, 1366, 618, 1492)], [(2104, 709, 2218, 823), (661,
1542, 837, 1718)], [(1805, 1403, 1985, 1583), (1936, 1781, 2005, 1850)]]
```

In [6]:
```
#Step 5: Dictionaries
zipimgs = {}

for i in range(len(pageslist)):
    zipimgs[pageslist[i].filename] = {}
    zipimgs[pageslist[i].filename]["Image"] = pillimg[i]
    zipimgs[pageslist[i].filename]["Text"] = textlist[i]
    zipimgs[pageslist[i].filename]["Face Coordinates"] = facecoor[i]
```

In [7]:
```
#Step 6: Extracting the faces
faceslist = []

for i in range(len(pageslist)):
    img = zipimgs[pageslist[i].filename]["Image"]
    facescrop = []
    for x1, y1, x2, y2 in zipimgs[pageslist[i].filename]["Face Coordinates"]:
        face = img.crop((x1,y1,x2,y2))
        facescrop.append(face)
    faceslist.append(facescrop)
```

In [8]:
```
#Step 7: appending faces

for i in range(len(pageslist)):
    zipimgs[pageslist[i].filename]["Faces"] = faceslist[i]
    print(len(zipimgs[pageslist[i].filename]["Faces"]))
```

```
6
5
2
2
```

In [9]:
```
#Step 8: Creating the Thumbnails

thumbnails = []

for i in range(len(pageslist)):
    whitetext = PIL.Image.new("RGB", (1000, 200 + 200*(int(len(zipimgs[pageslist
[i].filename]["Faces"])/5)+1)), color = 0)
    drawing = ImageDraw.Draw(whitetext)
    drawing.rectangle((0,0, 1000,200),fill = "white")
    drawing.text((200, 75), "Results found in file {}".format(pageslist[i].filenam
e), fill = "black")
    if len(zipimgs[pageslist[i].filename]["Faces"]) == 0:
        drawing.rectangle((0,0, 1000,400),fill = "white")
        drawing.text((200, 275), "But there were no faces in that file!", fill = "b
lack")
    else:
        xc = 0
        yc = 200
        for z in zipimgs[pageslist[i].filename]["Faces"]:
            tempi = z.resize((200,200))
            whitetext.paste(tempi, (xc, yc))
            xc = xc +200
            if xc == 1000:
                yc = yc + 200
                xc = 0
    thumbnails.append(whitetext)
```

In [10]:
```
#Step 9: Appending the thumbnails

for i in range(len(pageslist)):
    zipimgs[pageslist[i].filename]["Thumbnails"] = thumbnails[i]
    display(zipimgs[pageslist[i].filename]["Thumbnails"])
```

Results found in file a-0.png

Results found in file a-1.png

Results found in file a-2.png

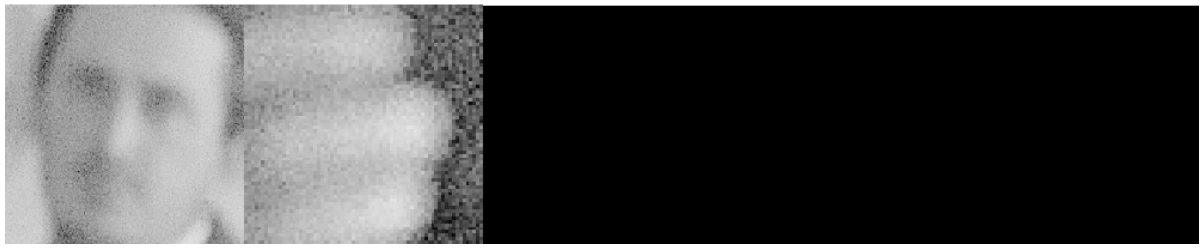
Results found in file a-3.png

In [11]:
```
#Step 10: Creating the word function
def WORDNFACES(wrd, dictionary):
    #With a word and a dictionary with faces and text, the function returns a list
with the "thumbnails" of those pages that contain the word
    thumblist = []
    for i in list(dictionary.keys()):
        if wrd in dictionary[i]["Text"]:
            thumblist.append(dictionary[i]["Thumbnails"])
    return thumblist
```

In [12]:
```
#Step 11: Testing
test12 = WORDNFACES("Christopher", zipimgs)
for i in test12:
    display(i)
```

Results found in file a-0.png

Results found in file a-3.png



In [13]:
```
#Step 12: Turning into functions
def ZIPOBJECT(zipfilelocation):
    #creates the initial zip object
    zipobject = zipfile.ZipFile(zipfilelocation)
    return zipobject

def FILENAMESLIST(zipobject):
    #create a list with filenames
    nameslist = zipobject.infolist()
    return nameslist
```

In [14]:
```
def IMGSLIST(zipobject):
    #Generates the list of pillimages from a zip object
    nameslist = FILENAMESLIST(zipobject)
    pillimg = []
    for i in nameslist:
        img = Image.open(zipobject.open(i))
        pillimg.append(img)
    return pillimg

def TEXTLIST(pillimg):
    #From a list of pillimages, return a list with the text from those images
    textlist = []
    for i in range(len(pillimg)):
        intcopy = pillimg[i]
        intcopy2 = intcopy.copy()
        intcopy2 = intcopy2.convert("1")
        text = pytesseract.image_to_string(intcopy2)
        textlist.append(text)
    return textlist

def FACECOORLIST(pillimg):
    #From a list of pillimages creates a list of tuples with the coordinates of the
faces in the picture
    facecoor = []
    for i in range(len(pillimg)):
        intcopy = pillimg[i]
        intcopy2 = intcopy.copy()
        open_cv_image = np.array(intcopy2)
        gray = cv.cvtColor(open_cv_image, cv.COLOR_BGR2GRAY)
        faces = face_cascade.detectMultiScale(gray, scaleFactor =1.3, minNeighbors
= 5)
        if faces == ():
            faces2 = []
        else:
            faces = faces.tolist()
            faces2 = []
            for i in faces:
                x1, y1, x2, y2 = i[0], i[1], i[0] + i[2], i[1] + i[3]
                coor = (x1,y1,x2,y2)
                faces2.append(coor)
        facecoor.append(faces2)
    return facecoor



def DICTIONARYSTEP1(pillimg, textlist, facecoor, nameslist):
    #Generates a dictionary with the images, text and coordinates of the pictures i
n a zipfile
    zipimgs = {}
    for i in range(len(nameslist)):
        zipimgs[nameslist[i].filename] = {}
        zipimgs[nameslist[i].filename]["Image"] = pillimg[i]
        zipimgs[nameslist[i].filename]["Text"] = textlist[i]
        zipimgs[nameslist[i].filename]["Face Coordinates"] = facecoor[i]
    return zipimgs
```

In [16]:
```
def FACESLIST(dictionarystep1, nameslist):
    #Using the dictionary from step 1 and the list of names of the zipfile, creates
the cropped images and adds them to the dictionary
    faceslist = []
    for i in range(len(nameslist)):
        img = dictionarystep1[nameslist[i].filename]["Image"]
        facescrop = []
        for x1, y1, x2, y2 in dictionarystep1[nameslist[i].filename]["Face Coordina
tes"]:
            face = img.crop((x1,y1,x2,y2))
            facescrop.append(face)
        faceslist.append(facescrop)
    return faceslist



def DICTIONARYSTEP2(dictionarystep1, nameslist):
    #Adds the faces to the dictionary
    faceslist = FACESLIST(dictionarystep1, nameslist)
    for i in range(len(nameslist)):
        dictionarystep1[nameslist[i].filename]["Faces"] = faceslist[i]


def THUMBLIST(dictionarystep2, nameslist):
    #With a dictionary on step 2 and a names list, creates the thumbnails
    thumbnails = []
    for i in range(len(nameslist)):
        whitetext = PIL.Image.new("RGB", (1000, 200 + 200*(int(len(dictionarystep2
[nameslist[i].filename]["Faces"])/5)+1)), color = 0)
        drawing = ImageDraw.Draw(whitetext)
        drawing.rectangle((0,0, 1000,200),fill = "white")
        drawing.text((200, 75), "Results found in file {}".format(nameslist[i].file
name), fill = "black")
        if len(dictionarystep2[nameslist[i].filename]["Faces"]) == 0:
            drawing.rectangle((0,200, 1000,400),fill = "white")
            drawing.text((200, 275), "But there were no faces in that file!", fill
= "black")
        else:
            xc = 0
            yc = 200
            for z in dictionarystep2[nameslist[i].filename]["Faces"]:
                tempi = z.resize((200,200))
                whitetext.paste(tempi, (xc, yc))
                xc = xc +200
                if xc == 1000:
                    yc = yc + 200
                    xc = 0
        thumbnails.append(whitetext)
    return thumbnails

def DICTIONARYSTEP3(dictionarystep2, nameslist):
    #Adds the thumbnails to the dictionary
    thumbnails = THUMBLIST(dictionarystep2, nameslist)
    for i in range(len(nameslist)):
        dictionarystep2[nameslist[i].filename]["Thumbnails"] = thumbnails[i]
```

In [17]:
```
#Working on the Large Zip File
large = ZIPOBJECT("readonly/images.zip")
largenameslist =  FILENAMESLIST(large)
```

In [18]:
```
pillimg2 = IMGSLIST(large)
```

In [19]:
```
textlist2 = TEXTLIST(pillimg2)
```

In [20]:
```
facecoor2 = FACECOORLIST(pillimg2)
```

In [21]:
```
dict1 = DICTIONARYSTEP1(pillimg2, textlist2, facecoor2, largenameslist)
```

In [22]:
```
DICTIONARYSTEP2(dict1, largenameslist)
```

In [23]:
```
DICTIONARYSTEP3(dict1, largenameslist)
```

In [24]:
```
test13 = WORDNFACES("Mark", dict1)
for i in test13:
    display(i)
```

Results found in file a-0.png

Results found in file a-1.png

Results found in file a-10.png

But there were no faces in that file!

Results found in file a-13.png

Results found in file a-2.png

```
Results found in file a-3.png
```

```
Results found in file a-8.png

But there were no faces in that file!
```

In [ ]: