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:

```
Results found in file a-0.png
```

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

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!):

```
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!
```

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

In [2]:

```
import math
import zipfile
from PIL import Image, ImageOps, ImageDraw
import pytesseract
import cv2 as cv
import numpy as np

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

# the rest is up to you!
```

```
Help on class ZipFile in module zipfile:

class ZipFile(builtins.object)
 |  ZipFile(file, mode='r', compression=0, allowZip64=True, compresslevel=
None)
 |
 |  Class with methods to open, read, write, close, list zip files.
 |
 |  z = ZipFile(file, mode="r", compression=ZIP_STORED, allowZip64=True,
 |              compresslevel=None)
 |
 |  file: Either the path to the file, or a file-like object.
 |        If it is a path, the file will be opened and closed by ZipFile.
 |  mode: The mode can be either read 'r', write 'w', exclusive create
'x',
 |        or append 'a'.
 |  compression: ZIP_STORED (no compression), ZIP_DEFLATED (requires zli
b),
 |               ZIP_BZIP2 (requires bz2) or ZIP_LZMA (requires lzma).
```

In [3]:

```
def extract_from_zip(file_name):
    images = {}
    myfile = zipfile.ZipFile(file_name)
    for i in myfile.infolist():
        file = myfile.open(i)
        img = Image.open(file).convert('RGB')
        images[i.filename] = {"pil-image":img}
    return images
```

In [4]:

```
#Extracting all the images from the zip file
all_images = extract_from_zip("readonly/small_img.zip")
```

In [5]:

```
print(all_images)
```

```
{'a-0.png': {'pil-image': <PIL.Image.Image image mode=RGB size=3600x6300 at
0x7F07A064B668>}, 'a-1.png': {'pil-image': <PIL.Image.Image image mode=RGB s
ize=3600x6300 at 0x7F07A3244BA8>}, 'a-2.png': {'pil-image': <PIL.Image.Image
image mode=RGB size=3600x6300 at 0x7F07A32440F0>}, 'a-3.png': {'pil-image':
<PIL.Image.Image image mode=RGB size=7200x6300 at 0x7F07A33109B0>}}
```

In [14]:

```
def exract_data(file_name):
    #text
    for i in file_name.keys():
        text = pytesseract.image_to_string(file_name[i]["pil-image"])
        file_name[i]["text"] = text

    #faces
    for i in file_name.keys():

        cv_image = np.array(file_name[i]['pil-image'])
        img_g = cv.cvtColor(cv_image, cv.COLOR_BGR2GRAY)

        faces_bounding_boxes = face_cascade.detectMultiScale(img_g, 1.3, 5)
        file_name[i]['faces'] = []

        for x,y,w,h in faces_bounding_boxes:
            face = file_name[i]['pil-image'].crop((x,y,x+w,y+h))
            file_name[i]['faces'].append(face)
```

In [7]:

```
exract_data(all_images)
```

In [8]:

```
print(all_images['a-0.png']['faces'])
```

```
[<PIL.Image.Image image mode=RGB size=280x280 at 0x7F07A324F860>, <PIL.Imag
e.Image image mode=RGB size=217x217 at 0x7F07A324FE80>, <PIL.Image.Image ima
ge mode=RGB size=198x198 at 0x7F07A05E1208>, <PIL.Image.Image image mode=RGB
size=213x213 at 0x7F07A0731358>, <PIL.Image.Image image mode=RGB size=197x19
7 at 0x7F07A325E0B8>, <PIL.Image.Image image mode=RGB size=276x276 at 0x7F07
A325E128>]
```

In [9]:

```
for img_name in all_images.keys():
    for face in all_images[img_name]['faces']:
        face.thumbnail((100,100),Image.ANTIALIAS)

print(all_images['a-0.png']['faces'])
```

```
[<PIL.Image.Image image mode=RGB size=100x100 at 0x7F07A324F860>, <PIL.Imag
e.Image image mode=RGB size=100x100 at 0x7F07A324FE80>, <PIL.Image.Image ima
ge mode=RGB size=100x100 at 0x7F07A05E1208>, <PIL.Image.Image image mode=RGB
size=100x100 at 0x7F07A0731358>, <PIL.Image.Image image mode=RGB size=100x10
0 at 0x7F07A325E0B8>, <PIL.Image.Image image mode=RGB size=100x100 at 0x7F07
A325E128>]
```

In [10]:

```
def search_for(keyword,file_name):
    for img_name in file_name.keys():

        if (keyword in file_name[img_name]['text']):

            if(len(file_name[img_name]['faces']) != 0):
                print("Result found in file {}".format(img_name))
                h = math.ceil(len(file_name[img_name]['faces'])/5)
                contact_sheet=Image.new('RGB',(500, 100*h))
                xc = 0
                yc = 0

                for img in file_name[img_name]['faces']:
                    contact_sheet.paste(img, (xc, yc))

                    if xc + 100 == contact_sheet.width:
                        xc = 0
                        yc += 100
                    else:
                        xc += 100

                display(contact_sheet)
            else:
                print("Result found in file {} \nBut there were no faces in that file\n\n".
    return
```

In [11]:

```
search_for('Chris',all_images)
```

```
Result found in file a-0.png
```

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

In [15]:

```
big_images = extract_from_zip("readonly/images.zip")
exract_data(big_images)

for img_name in big_images.keys():
    for face in big_images[img_name]['faces']:
        face.thumbnail((100,100),Image.ANTIALIAS)

print(big_images['a-0.png']['faces'])
```

```
[<PIL.Image.Image image mode=RGB size=100x100 at 0x7F072C5737F0>, <PIL.Imag
e.Image image mode=RGB size=100x100 at 0x7F072C573C18>, <PIL.Image.Image ima
ge mode=RGB size=100x100 at 0x7F072C5736D8>, <PIL.Image.Image image mode=RGB
size=100x100 at 0x7F072C573748>, <PIL.Image.Image image mode=RGB size=100x10
0 at 0x7F072C5737B8>, <PIL.Image.Image image mode=RGB size=100x100 at 0x7F07
2C573F98>]
```

In [17]:

```
search_for('Mark',big_images)
```

Result found in file a-0.png

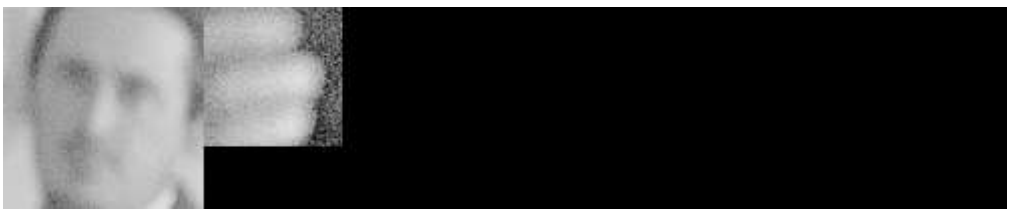
Result found in file a-1.png

Result found in file a-10.png
But there were no faces in that file

Result found in file a-13.png

Result found in file a-2.png

Result found in file a-3.png



Result found in file a-8.png
But there were no faces in that file