

TP 2 - Lecture automatique de chiffre par analyse d'image

I - Présentation du TP

II - Prise en main de l'environnement

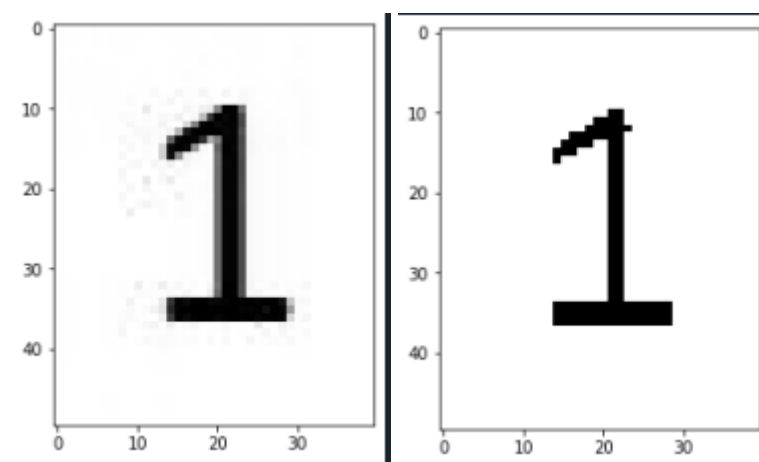
III - Travail préparatoire

- 1)
- 2) Écrire une méthode `binarisation(self, S)` à la classe `Image` qui permet de passer d'une image codée sur 256 valeurs à une image avec seulement deux valeurs (0 ou 255).

On parcourt l'image pixel par pixel en observant si l'intensité est supérieur ou inférieur au seuils `S`

```
def binarisation(self, S):  
    im_bin = Image()  
    im_bin.set_pixels(np.zeros((self.H, self.W), dtype=np.uint8))  
    for i in range(self.H):  
        for j in range(self.W):  
            if self.pixels[i][j] > S:  
                im_bin.pixels[i][j] = 255  
    return im_bin
```

image obtenue:



3) Écrire une méthode localisation(self) à la classe Image calculant et retournant l'image recadrée sur le chiffre à identifier.

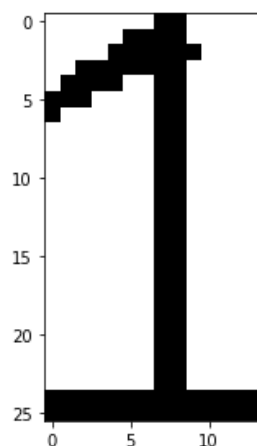
on initialise les valeurs des "côtés" du rectangle, on parcourt l'image, lorsqu'on trouve un pixel noir, si il le faut on met a jour les valeurs initialisé au début.

Lorsqu'on a parcouru tous les pixels, on crée une nouvelle image avec les nouvelles dimensions et on l'a remplie avec les pixels de l'image de base.

```
self=self.binarisation(70)
    cmin=0
    cmax=self.W
    lmin=0
    lmax=self.H
    for i in range(self.W):
        for k in range(self.H):
            if self.pixels[k][i]==0:
                if i<lmax:
                    lmax=i
                if i>lmin:
                    lmin=i
            if self.pixels[k][i]==0:
                if k<cmax:
                    cmax=k
                if k>cmin:
                    cmin=k

    im_bin = Image()
    im_bin.set_pixels(np.zeros((cmin-cmax,lmin-lmax),
dtype=np.uint8)

    for i in range (im_bin.H):
        for k in range (im_bin.W):
            im_bin.pixels[i][k]=self.pixels[i+cmax][k+lmax]
    return(im_bin)
```

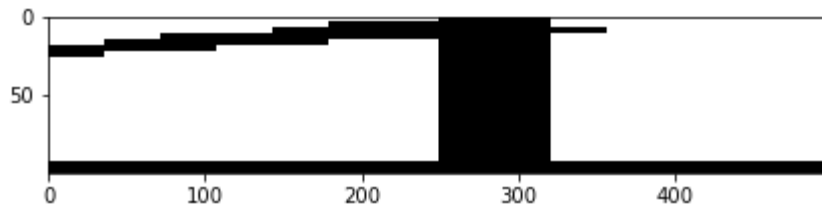


IV - Reconnaissance automatique de chiffre

- 1) binarisation fonctionne bien
- 2) localisation fonctionne bien
- 3) Ajouter à la classe Image la méthode **resize(self,new_H,new_W)** qui redimensionne l'image à la taille voulue et renvoie un autre objet de type Image en sortie.

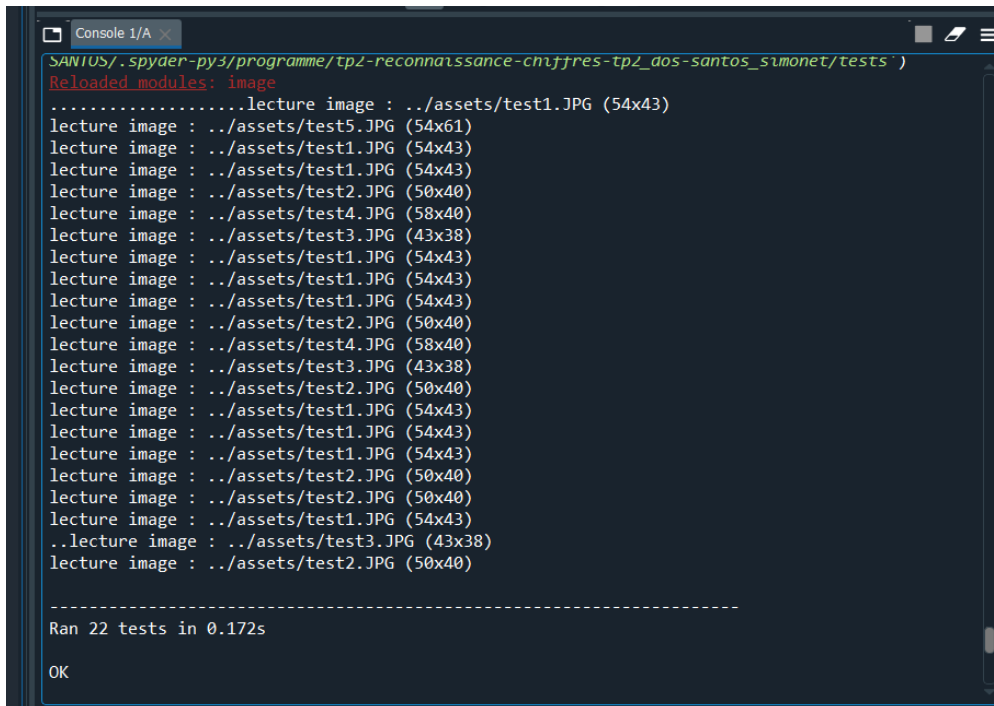
On utilise la fonction donnée dans le sujet pour construire la fonction resize

```
def resize(self, new_H, new_W):  
    im_res=Image()  
    im_res_pixel=resize(self.pixels, (new_H,new_W), 0)  
    im_res.set_pixels(np.uint8(im_res_pixel*255))  
    return(im_res)
```



- 4) Ajouter à la classe Image, la méthode **similitude(self, image)** qui mesure la similitude par corrélation d'images

```
def similitude(self, im):  
    tot=self.H*self.W  
    simi=0  
    for i in range(self.H):  
        for k in range (self.W):  
            if self.pixels[i][k]==im.pixels[i][k]:  
                simi+=1  
    return(sim_i/tot)
```



```
SAVIUS/.spyder-py3/programme/tp2-reconnaissance-chiffres-tp2-dos-santos-simonet/tests )
Reloaded modules: image
.....lecture image : ../assets/test1.JPG (54x43)
lecture image : ../assets/test5.JPG (54x61)
lecture image : ../assets/test1.JPG (54x43)
lecture image : ../assets/test1.JPG (54x43)
lecture image : ../assets/test2.JPG (50x40)
lecture image : ../assets/test4.JPG (58x40)
lecture image : ../assets/test3.JPG (43x38)
lecture image : ../assets/test1.JPG (54x43)
lecture image : ../assets/test1.JPG (54x43)
lecture image : ../assets/test1.JPG (54x43)
lecture image : ../assets/test2.JPG (50x40)
lecture image : ../assets/test4.JPG (58x40)
lecture image : ../assets/test3.JPG (43x38)
lecture image : ../assets/test2.JPG (50x40)
lecture image : ../assets/test1.JPG (54x43)
lecture image : ../assets/test1.JPG (54x43)
lecture image : ../assets/test1.JPG (54x43)
lecture image : ../assets/test2.JPG (50x40)
lecture image : ../assets/test2.JPG (50x40)
lecture image : ../assets/test1.JPG (54x43)
..lecture image : ../assets/test3.JPG (43x38)
lecture image : ../assets/test2.JPG (50x40)

-----
Ran 22 tests in 0.172s

OK
```

- 5) Dans le fichier reconnaissance.py, écrire la fonction reconnaissance_chiffre(image, liste_modeles, S) qui va effectuer la reconnaissance de chiffre sur l'image image donnée en entrée de la fonction.

```
def reconnaissance_chiffre(image, liste_modeles, S):
    image=image.binarisation(S)
    image=image.localisation()
    x=0
    indice=0
    for k in range (len(liste_modeles)):
        liste_modeles[k]=liste_modeles[k].binarisation(S)
        liste_modeles[k]=liste_modeles[k].localisation()
        liste_modeles[k]=liste_modeles[k].resize(image.H,image.W)
        y=image.similitude(liste_modeles[k])
        if y>x:
            x=y
            indice=k
    return(indice)
```



on a fait le test reconnaissance tout a été vérifié, voici le résultat

```
...lecture image : ../assets/_0.png (32x22)
lecture image : ../assets/_1.png (32x18)
lecture image : ../assets/_2.png (32x20)
lecture image : ../assets/_3.png (32x20)
lecture image : ../assets/_4.png (32x24)
lecture image : ../assets/_5.png (32x20)
lecture image : ../assets/_6.png (32x21)
lecture image : ../assets/_7.png (32x21)
lecture image : ../assets/_8.png (32x22)
lecture image : ../assets/_9.png (32x22)
lecture image : ../assets/test2.JPG (50x40)
lecture image : ../assets/_0.png (32x22)
lecture image : ../assets/_1.png (32x18)
lecture image : ../assets/_2.png (32x20)
lecture image : ../assets/_3.png (32x20)
lecture image : ../assets/_4.png (32x24)
lecture image : ../assets/_5.png (32x20)
lecture image : ../assets/_6.png (32x21)
lecture image : ../assets/_7.png (32x21)
lecture image : ../assets/_8.png (32x22)
lecture image : ../assets/_9.png (32x22)
lecture image : ../assets/test1.JPG (54x43)
lecture image : ../assets/_0.png (32x22)
lecture image : ../assets/_1.png (32x18)
lecture image : ../assets/_2.png (32x20)
lecture image : ../assets/_3.png (32x20)
lecture image : ../assets/_4.png (32x24)
lecture image : ../assets/_5.png (32x20)
lecture image : ../assets/_6.png (32x21)
lecture image : ../assets/_7.png (32x21)
lecture image : ../assets/_8.png (32x22)
lecture image : ../assets/_9.png (32x22)
lecture image : ../assets/test3.JPG (43x38)

-----
Ran 3 tests in 0.148s
```