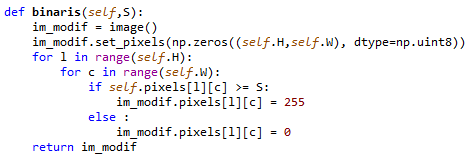
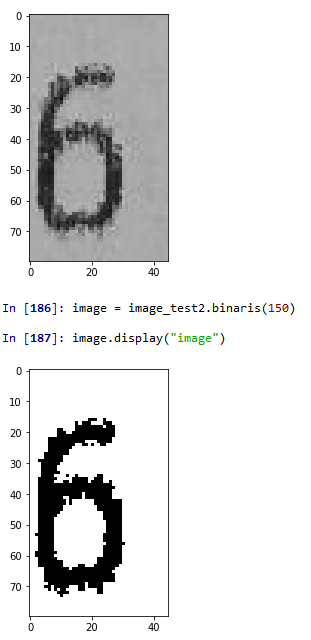
**Compte-rendu du TP1**

**Partie 1: Travail Préparatoire**

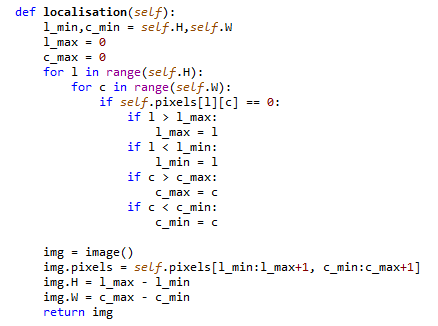
Pour la première question, binaris va vérifier une à une chaque pixel de l’image et comparer leur valeur au seuil. En fonction de cette valeur, le programme va rendre le pixel blanc ou noir.



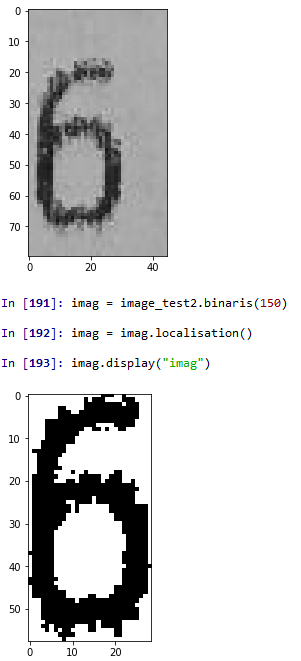
Binaris renvoie alors ceci :



Localisation va scanner l’image pixel par pixel, si le pixel est blanc, il ne fait rien, si le pixel est noir, il va comparer les coordonnées du pixel en question avec les quatre valeurs que l’on recherche pour vérifier s’il se trouve dans le rectangle formé par ces quatre valeurs. Si ce n’est pas le cas, le programme va ajuster ces valeurs.



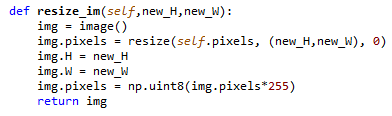
Localisation renvoie alors ceci, on peut bien voir que l’image est bien découpée, et qu’aucun pixel ne manque



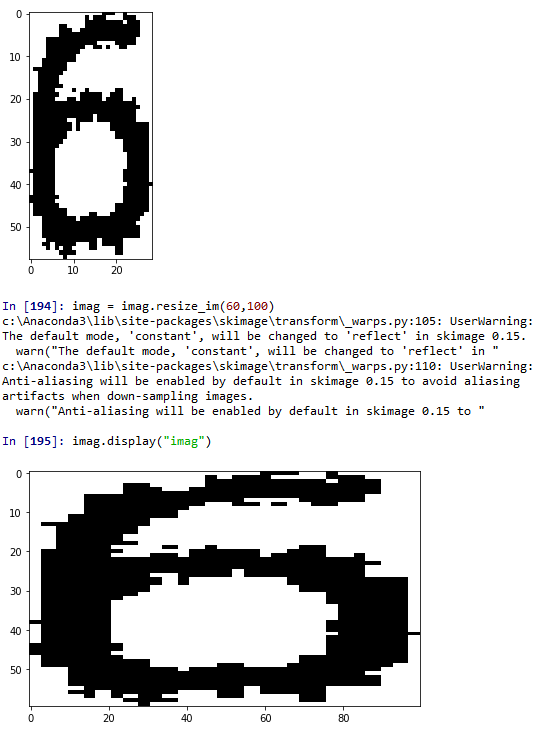
**Partie 3: Reconnaissance automatique de chiffres**

**Question 3-4 :**

Resize\_im se sert de la fonction resize qui existe déjà. On créé une nouvelle image avec les caractéristiques demandées. La dernière ligne permet aux pixels de la nouvelle image d’avoir une valeur de 0 ou 255 au lieu de 0 ou 1

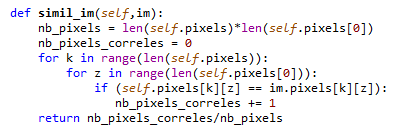


Le programme renvoie alors ceci :



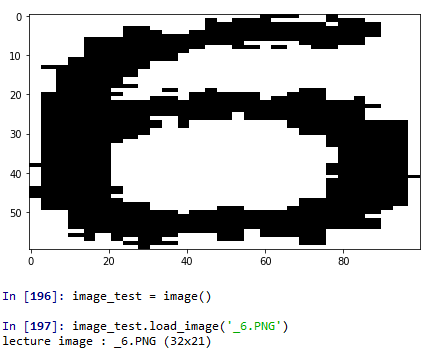
**Question 3-5 :**

Simil\_im va parcourir chaque pixel des deux images à comparer, et va compter le nombre de pixels identiques et renvoyer la proportion de pixels identiques des images



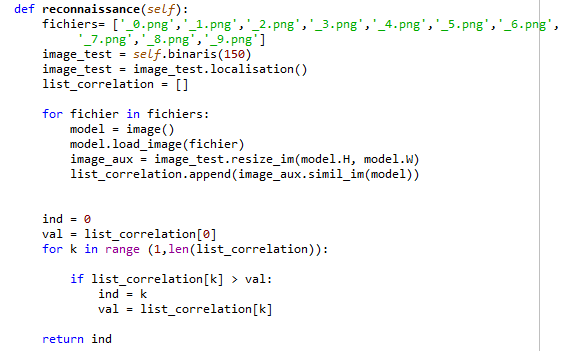
Le programme renvoie une valeur de corrélation de 0.76 entre l’image localisée et l’image 6 exemple, les résultats renvoyés par le programme sont cohérents

**C:\Users\forestay\AppData\Local\Microsoft\Windows\INetCache\Content.Word\Capture resultat simil 2.PNGC:\Users\forestay\AppData\Local\Microsoft\Windows\INetCache\Content.Word\Capture resultat simil 3.PNG**

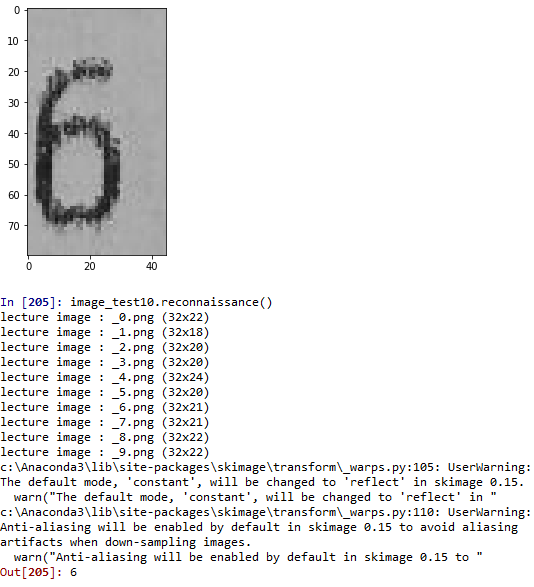


**Question 3-6 :**

Reconnaissance va récupérer les images de chaque chiffre, puis va calculer le coefficient de corrélation entre l’image originale et chacune de ces images et les renvoyer dans une liste. Il va ensuite renvoyer l’indice du coefficient le plus élevé de la liste. (Un coefficient k de cette liste correspond au coefficient de corrélation avec l’image du chiffre k)

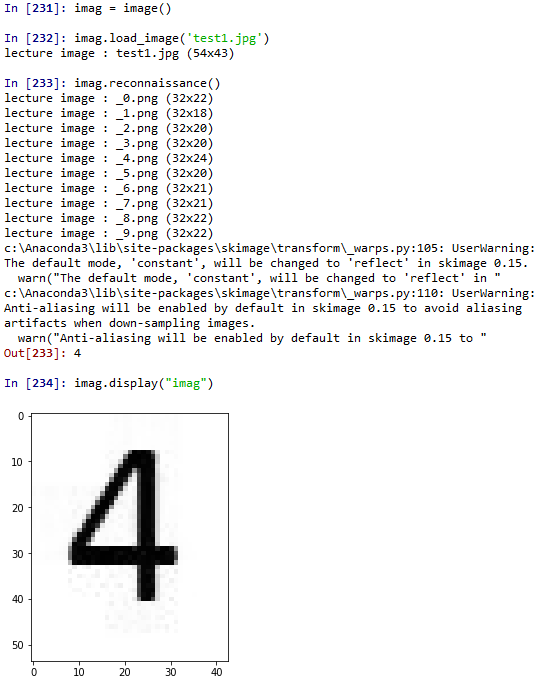


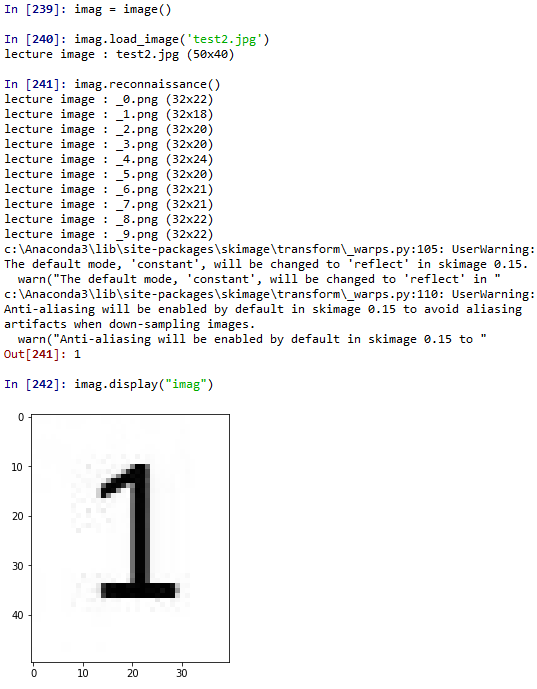
On peut bien voir (en bas Out[205]) que le programme est en effet capable de reconnaitre le chiffre 6



**Partie 4 Tests :**

Le programme est en mesure de détecter les images autres que test10 (voir Out[233] et Out[241]):





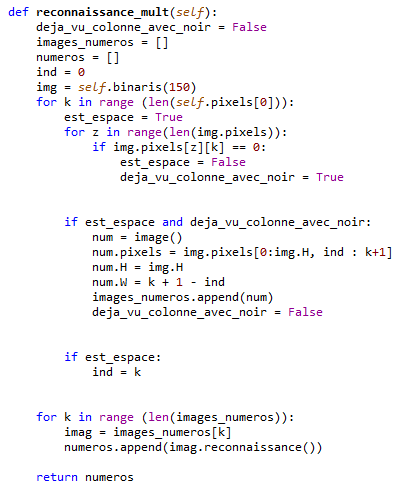
Le programme est en mesure de détecter correctement les images pour un seuil (pour binaris) entre ~60 et ~160

**Partie 5 : Pour les étudiants en avance**

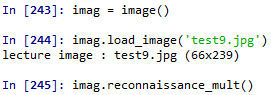
Reconnaissance\_mult fonctionne en deux parties :

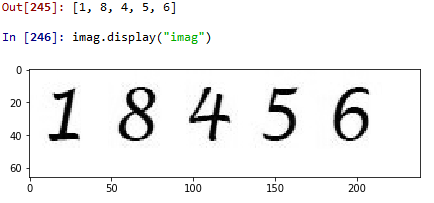
-la première permet de séparer les différents chiffres présents sur l’image. Le programme va scanner toutes les colonnes, s’il n’y a pas de pixels noir sur la ligne, il considère cela comme un espace, et s’en sert pour séparer les différents chiffres. Le programme renvoie une liste avec les images des différents chiffres.

-La deuxième partie va utiliser reconnaissance sur chaque image de la liste précédemment créée puis renvoyer une liste avec les valeurs données par chaque appel de reconnaissance.



Le programme renvoie alors ceci :





On peut voir (Out[245]) que le résultat est bon.