

1 Représentation des nombres

Q1.

base 2	Valeur si la variable stockée est de type <i>uint8</i>	Valeur si la variable stockée est de type <i>int8</i>
<u>0101 1010</u>		
<u>1001 1111</u>		

Q2.

Q3.

Q4.

Q5.

2 Dissimulation d'image

```
"""
```

IMPORTATION DES BIBLIOTHEQUES

```
"""
```

```
import matplotlib.image as mping
import matplotlib.pyplot as plt
import numpy as np
```

```
"""
```

AFFICHAGE DES IMAGES BRUTES

```
"""
```

```
print('la photographie brute est :')
im_orig_1=mping.imread('1_photographie.png',
plt.imshow(im_orig_1)
plt.show()
print('le type initial du vert de la photographie est :')
mot=im_orig_1[20,30,1]
print(mot)
"""
```

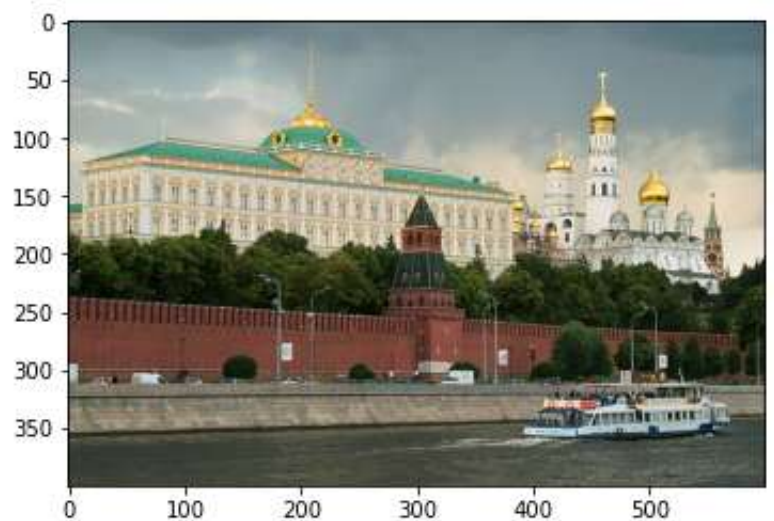
FONCTION DE CONVERSION DES FLOTTANTS EN ENTIERS

```
"""
```

```
def conv_flot_entier(cheminimage):
    #LECTURE DE L'IMAGE D'ORIGINE
    im_orig=mping.imread(cheminimage)
    #CREATION D'UNE COPIE
    im_copie=im_orig.copy()
    #CONVERSION
    if im_copie.dtype==np.float32:
        im_copie=(im_orig*255).astype(np.uint8)
    return im_copie
```

#TEST DE LA FONCTION conv_flot_entier(cheminimage)

```
im_flot=conv_flot_entier('1_photographie.png')
print('le vert de la photographie transformé en uint8 :')
mot=im_flot[20,30,1]
print(mot)
```



le type initial du vert de
la photographie est :

0.5803922

le vert de la photographie
transformé en uint8 :

148

"""

SUPPRESSION DES BITS DE POIDS FAIBLE

"""

```
def supp_poids_faible(cheminimage):  
    #LECTURE ET CONVERSION DE L'IMAGE D'ORIGINE  
    im_orig=conv_flot_entier(cheminimage)  
    #SUPPRESSION DES BITS DE POIDS FAIBLE
```

À compléter q1.

```
#TEST DE LA FONCTION supp_poids_faible(cheminimage)  
im_4bits=supp_poids_faible('1_photographie.png')  
print('le vert de la photographie est maintenant :')  
mot=im_4bits[20,30,1]  
print(mot)  
print('le vert de la photographie transformé en binaire :')  
mot=bin(mot) ←  
print(mot)  
print('attention :')  
print(type(mot))  
plt.imshow(im_4bits)  
plt.show()
```

le vert de la photographie est maintenant :

144

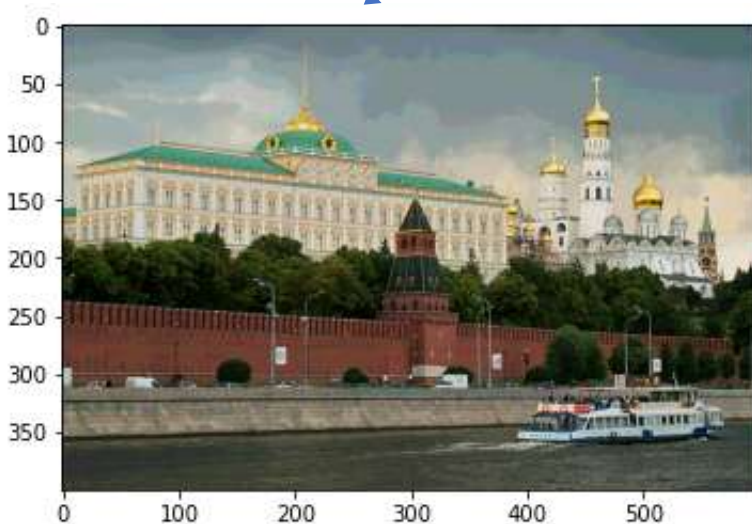
le vert de la photographie transformé en binaire :

0b10010000

attention :

<class 'str'>

À compléter q2.



"""

CONCATENATION

"""

```
def concatener_pf(ch1,ch2):
```

À compléter q3.

```
#TEST DE LA FONCTION concatener_pf(ch1,ch2)
print('la concaténation des bits de poids fort renvoie :')
print(concatener_pf("0b10010000","0b11100000"))
```

la concaténation des bits de poids
fort renvoie :

0b10011110

"""

TRANSFORMATION D'UNE CHAINE DE CARACTERE BINAIRE EN INT

"""

```
def entier(chaine):
    #La chaine est au format "0bxxxxxxx"
```

À compléter q4.

```
#TEST DE LA FONCTION entier(chaine)
print('la transformation de la chaine au format \
      0bxxxxxxx en entier renvoie:')
print(entier("0b10011110"))
```

la transformation de la chaine au format
0bxxxxxxx en entier renvoie:

158

"""

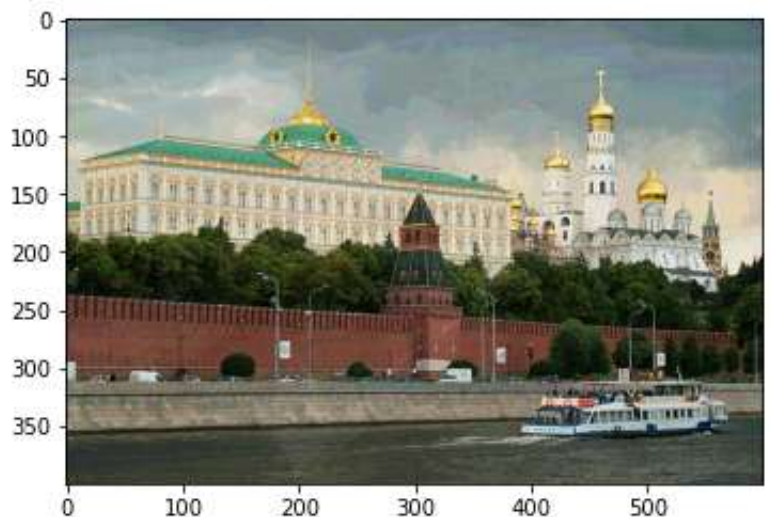
CAMOUFLAGE

"""

```
def camoufler(cheminimage1,cheminimage2):  
    #cheminimage1 pointe l'image touristique  
    #cheminimage2 pointe l'image à camoufler  
    #les 2 images ont les mêmes dimensions en  
    #hauteur et en largeur.  
    #LECTURE ET CONVERSION DES IMAGES D'ORIGINE  
    im_touriste=conv_flot_entier(cheminimage1)  
    im_camouflee=conv_flot_entier(cheminimage2)
```

À compléter q5.

```
#TEST DE LA FONCTION camoufler(cheminimage1,cheminimage2)  
#ET ENREGISTREMENT DE L'IMAGE  
print('la photographie camouflée est :')  
im_orig_cam=camoufler('1_photographie.png','2_image_secrete.png')  
plt.imshow(im_orig_cam)  
mping.imsave("3_je_vous_ai_eu_les_mechants_ha_ha.png",im_orig_cam)  
plt.show()
```



"""

ECLATEMENT

"""

```
def eclater(ch):
```

À compléter q6.

Remarque : pour la même raison que pour la concaténation, seule la partie #ECLATEMENT est requise sur la copie.

```
#TEST DE LA FONCTION eclater(ch)
```

```
print("l'éclatement de chaine renvoie :")
```

```
print(eclater("0b00011000"))
```

"""

l'éclatement de chaine renvoie :

('0b00010000', '0b10000000')

EXTRACTION

"""

```
def extraire(cheminimage):
```

```
    #LECTURE ET CONVERSION DES IMAGES D'ORIGINE
```

```
    im_orig=conv_flot_entier(cheminimage)
```

```
    #CONVERSION DES IMAGES
```

À compléter q7.

```
#TEST DE LA FONCTION extraire(cheminimage)
```

```
print("l'extraction renvoie :")
```

```
im_tour,im_bat=extraire("3_je_vous_ai_eu_les_mechants_ha_ha.png")
```

```
plt.imshow(im_bat)
```

```
plt.show()
```

```
plt.imshow(im_tour)
```

```
plt.show()
```