```
Exercice 1 :
from PIL import Image
import numpy as np

tab = Image.open("tours.JPG")
tab = np.array(tab,dtype = "uint8")

"""exercice 1 : négatif"""
tab_new1=np.copy(tab)
tab_new1=255-tab
#photo1=Image.fromarray(tab_new1)
#photo1.show()

"""exercice 1 : filtre rouge : """
tab_new2=np.copy(tab)
tab_new2[:,:,1:3]=0
#photo2=Image.fromarray(tab_new2)
#photo2.show()

"""exercice 1 : niveau de gris : """
l,c,p=np.shape(tab)
tab_new3=np.zeros((l,c),dtype="uint8")
tab_new3[:,:]=0.299*tab[:,:,0]+0.587*tab[:,:,1]+0.114*tab[:
,:,2]
#photo3=Image.fromarray(tab_new3)
#photo3.show()

"""exercice 1 : seuillage"""
"""seuillage version gris"""
l,c,p=np.shape(tab)#dimension du tableau
tab_new=np.zeros((l,c),dtype="uint8")
tab_new[:,:]=0.299*tab[:,:,0]+0.587*tab[:,:,1]+0.114*tab[:,
:,2]
mask=tab_new[:,:]>123
tab_new[mask]=255
tab_new[np.invert(mask)]=0
photo=Image.fromarray(tab_new)
photo.show()
"""seuillage vesrions couleur"""
mask1=tab[:,:,0]>123
mask2=tab[:,:,1]>123
mask3=tab[:,:,2]>123
mask=mask1+mask2+mask3
tab_new=np.copy(tab)
tab_new[mask]=255
tab_new[np.invert(mask)]=0
photo=Image.fromarray(tab_new)
photo.show()

"""exercice 1 : kernel"""
l,c,p=np.shape(tab)
tab_new3=np.zeros((l,c),dtype="float")
tab_new3[:,:]=0.299*tab[:,:,0]+0.587*tab[:,:,1]+0.114*tab[:
,:,2]
tab_new5=np.copy(tab_new3)
tab_new5[1:-1,1:-1]=abs(-tab_new3[0:-2,0:-2]+tab_new3[0:-
2,2:]-2*tab_new3[1:-1,0:-2]
+2*tab_new3[1:-1,2:]-tab_new3[2:,0:-2]+tab_new3[2:,2:])
tab_new5=np.array(tab_new5,dtype="uint8")
#photo5=Image.fromarray(tab_new5)
#photo5.show()


"""exercice 1 : symétrie axial"""
tab_new6=np.copy(tab)
tab_new6=tab[-1::-1,:,:]
#photo6=Image.fromarray(tab_new6)
#photo6.show()


"""exercice 1 : contour"""
tab_new7=np.copy(tab)
tab_new7[:50,:,:]=tab_new7[:,0:50,:]=tab_new7[-
49:,:,:]=tab_new7[:,-49:,:]=0
photo7=Image.fromarray(tab_new7)
photo7.show()
```

Exercice 2 :

```
from PIL import Image
import numpy as np
import time
"""question 1"""
tab1 = Image.open("personnage.png")
tab1 = np.array(tab1,dtype = "uint8")
tab2 = Image.open("carte.png")
tab2 = np.array(tab2,dtype = "uint8")
"""question2"""
print(tab1[100,100], tab1[95,100], tab1[100,95],
tab1[105,100], tab1[100,105])
"""question3"""
def distance(tab,i,j,pixel_ref):
    pixel=np.array(tab[i,j],dtype="int64")
    pixel_ref=np.array(pixel_ref,dtype="int64")
    vecteur_diff = np.array(pixel-pixel_ref)
    return
(vecteur_diff[0]**2+vecteur_diff[1]**2+vecteur_diff[2]**2)*
*0.5
"""question 4"""
tab2_new=np.copy(tab2)
def incrustation(tab1,tab2,condition) :
    l,c,p=np.shape(tab1)
    for i in range(l):
        for j in range(c):
            if distance(tab1,i,j,tab1[100,100])>condition:
                tab2_new[i,j]=tab1[i,j]
"""question5"""
t_debut=time.time()
incrustation(tab1,tab2,50)
t_fin=time.time()
duree=t_fin-t_debut
print(duree)
```

```
photo=Image.fromarray(tab2_new)
#photo.show()
"""question 6"""
pixel_ref=np.array(tab1[100,100],dtype="int64")
tab1_new=np.copy(tab1)
tab1_new=np.array(tab1_new,dtype="int64")
mask=((tab1_new[:,:,0]-pixel_ref[0])**2+(tab1_new[:,:,1]-
pixel_ref[1])**2+(tab1_new[:,:,2]-pixel_ref[2])**2)**0.5>50

"""question7"""
tab3_new=np.copy(tab2)
t_debut=time.time()
mask=((tab1_new[:,:,0]-pixel_ref[0])**2+(tab1_new[:,:,1]-
pixel_ref[1])**2+(tab1_new[:,:,2]-pixel_ref[2])**2)**0.5>50
tab3_new[mask]=tab1[mask]
t_fin=time.time()
duree=t_fin-t_debut
print(duree)
photo=Image.fromarray(tab3_new)
photo.show()
```