Devoir d'informatique

1 Représentation des nombres

On dispose d'un système d'exploitation ou les nombres sont représentables sur 10 bits.

Question 1 *Combien de nombres entiers peut-on représenter ?*

Question 2 *Quels sont le plus petit et le plus grand entier naturel représentables ?*

Question 3 *Quels sont le plus petit et le plus grand entier relatif représentables ?*

Question 4 *Convertir en base 2 le nombre suivant :* $(155)_{10}$.

Question 5 *Convertir en base 16 le nombre suivant :* $(155)_{10}$.

Question 6 *Convertir en base 10 le nombre suivant :* $(B3)_{16}$.

On dispose d'un système d'exploitation où les nombres sont représentables sur 4 bits.

Question 7 *Convertir en base 2 le nombre suivant :* $(-6)_{10}$.

2 Mesure du contraste

Question 8 *Préciser l'espace mémoire nécessaire pour stocker la valeur d'une composante, puis celle d'un pixel et enfin celle d'une image en* Mo *(*$=1000$ ko*) ou* Mio *(*$=1024$ kio*).*

Correction Une composante de couleur est un nombre compris entre 0 et 255. Il se stocke sur 1 octet.
Un pixel stocké au format RVB est composé de 3 nombres compris entre 0 et 255. Un pixel nécessite donc 3 octets de stockage.
Une image de 48 MPixels utilisera donc $3 \times 48 \times 10^6$ octets, c'est à dire 144 Mo.

Conversion en niveau de gris

Question 9 *Écrire une fonction* `Clinear(val)` *, qui prend en argument une valeur de l'espace non linéaire et qui renvoie la valeur linéarisée.*

Correction

```
def Clinear(val):
        if val<=0.04045:
                Clin=val/12.92
        else:
```

```
            Clin=((val+0.055)/1.055)**2.4
        return Clin
```

Question 10 *Écrire une fonction* `Y(pix)` *qui prend en argument une liste de trois valeurs correspondant à un pixel au format RVB et qui renvoie la valeur Y du niveau de gris dans l'espace non linéaire.*

Correction

```
def Y(pix):
        Ylin = 0.2126*pix[0] + 0.7152*pix[1] + 0.0722*pix[2]
        if Ylin<=0.0031308:
                Y=12.92*Ylin
        else:
                Y=1.055*Ylin**(1/2.4)-0.055
        return Y
```

Question 11 *Écrire une fonction* `NiveauxGris(I)` *prenant en argument une image* `I` *au format RVB et qui renvoie une image de même dimension en niveau de gris.*

Correction

```
def NiveauxGris(I):
        taillex,tailley,nb_couleurs = np.shape(I) # taille de l'image initiale
        Igris=np.zeros(taillex,tailley) # on crée l'image en niveaux de gris
        for x in range(taillex):
                for y in range(tailley):
                        Rlin=Clinear(I[x,y,0])
                        Vlin=Clinear(I[x,y,1])
                        Blin=Clinear(I[x,y,2])
                        Igris[x,y]=Y([Rlin,Vlin,Blin])
        return Igris
```

Question 12 *Écrire une fonction* `convolution(A,B)` *prenant en argument deux matrices de taille* $3x3$ *et qui renvoie la valeur du produit de convolution.*

Correction

```
def convolution(A,B):
    conv=0      # on initialise
    for i in range(3):
        for j in range(3):
            conv = conv + A[i,j]*B[i,j]
    return conv
```

Question 13 *Écrire une fonction* `contraste_pixel(I,i,j)` *prenant en argument une image* `I` *au format niveaux de gris et les coordonnées du pixel* (i, j) *qui renvoie la valeur du contraste défini précédemment par la quantité* c.

Correction

```
def contraste_pixel(I,i,j):
    Gx=np.array([[ -1, 0, 1],
                 [ -2, 0, 2],
                 [ -1, 0, 1]])
    Gy=np.array([[ -1, -2, -1],
                 [ 0, 0, 0],
                 [ 1, 2, 1]])
    conv1=convolution(I,Gx)
    conv2=convolution(I,Gy)
```

```
    c=int(np.sqrt(conv1**2+conv2**2))
    return c
```

Question 14 *Écrire une fonction* `contraste(I)` *prenant en argument une image* `I` *au format niveau de gris et qui renvoie la valeur du contraste de référence* c_{ref}.

Correction

```
def contraste(I):
        taillex,tailley = np.shape(I)    # taille de l'image
        cref=0
        for i in range(1,taillex-1):
                for j in range(1,tailley-1):    # on retire les pixels des bords
                        cref = cref + contraste_pixel(I,i,j)
        cref = cref / ((taillex-2)*(tailley-2))  # on fait la moyenne
        return cref
```

Question 15 *Écrire une fonction* `reglage` *, dont les arguments et les valeurs de retour sont à définir, répondant au comportement décrit en partant de la position 0 en pas. On supposera que le maximum de contraste existe.*

Correction

```
def reglage(I,val):
    '''
    I : image initiale
    val : position de l'objectif '''
    cref=contraste(I) # contraste de l'image initiale
    sens=1 # sens de déplacement de l'objectif (1 : sens + ; -1 : sens -)
    position_objectif(val+sens) # première itération pour savoir dans quel sens il faut ↙
        avancer
    Img1=prise()
    c=contraste(Img1)
    if c<cref: # si on est dans le mauvais sens, on repart de I
        sens = -sens
        c=cref

    while c<=cref:
        cref=c
        val = val+sens
        position_objectif(val)
        Img1=prise()
        c=contraste(Img1)

    return cref
```