INFORMATIQUE MP DEVOIR 2

CORRECTION

PARTIE A- RÉCUPÉRATION DES DONNÉES

1. Il faut 201 entiers à encoder, il faut donc au moins 8 bits : $2^8 = 256$.
2. Une mesure complète lors d'une heure (mesure + informations) prends 1 Mo. On dispose les stations sur une surface de 100m × 80m à raison d'une station par m^2. Si on fait des mesures pendant 12h, de quelle quantité de mémoire doit-on disposer sur le serveur ? Il faut $80 \times 100 \times 12 = 96000 Mo$
3. ```
   def repart(mesures):
       P,T=[],[]
       for mes in mesures :
           if mes[0]=='P':
               for valeur in mes[1]:
                   P.append(valeur)
           if mes[0]=='T':
               for valeur in mes[1]:
                   T.append(valeur)
           if mes[0]=='C':
               check = mes[1]
       return(P,T,check)
   ```
4. ```
   def checksum(mesures):
       P,T,check=repart(mesures)
       return(check == sum(P)+sum(T))
   ```

PARTIE B - COHÉRENCE DES DONNÉES

1. ```
   def somme_abs(t):
       S = 0
       for val in t:
           S += abs(val)
       return(S)
   ```
2. ```
   def trapeze(t):
       S=0
       for k in range(len(t)-1):
           S += 1*(t[k]+t[k+1])/2
       return(S/(len(t)-1=)
   ```
3. ```
   def coherent(P,T):
       return( abs(somme_abs(P)-2*trapeze(P))<1 and abs(somme_abs(T)-2*trapeze(T))<1 )
   ```

PARTIE C - BASE DE DONNÉES

1. Écrire une requête SQL donnant les noms des scientifiques du centre nord-est.

   ```
   SELECT nom FROM scientifique WHERE centre = 'nord-est'
   ```

2. Écrire une requête SQL renvoyant le nom du scientifique, le type de mesure et le fichier correspondant pour chaque mesure.

   ```
   SELECT nom,type,fichier FROM scientifique JOIN mesure ON iden=idnom
   ```

3. Écrire une requête SQL renvoyant le nombre de mesures de chaque type effectuées.

   ```
   SELECT type,COUNT(*) FROM  mesure GROUP BY type
   ```

4. Écrire une requête SQL donnant le nom du scientifique ayant fait le plus de mesures de type détection, avec ce nombre.
   ```

```
SELECT nom,max(nb)   FROM  scientifique
JOIN (SELECT idnom,COUNT(*) as nb FROM mesure WHERE type = 'd' GROUP BY idnom)
ON idnom=iden
```

PARTIE D - CONSTRUCTION D'UNE IMAGE

1. Écrire une fonction `niveau(P,T)` qui renvoie la valeur `N` du niveau de gris selon se calcul.

```
def niveau(P,T):
   return(abs(P)/(abs(P)+abs(T)))
```

2. Écrire une fonction `image_gris(releve)` qui construit et renvoie cette image en niveau de gris.

```
def image_vide(n,p):
   return([[0 for j in range(p)] for i in range(n)])

def image_gris(releve):
   im = image_vide(100,80)
   for val in releve :
      x,y,P,T = val
      N = calcul(P,T)
      im[y][x]=N
   return(im)
```

3. Écrire une fonction `image_risques(releve)` qui construit et renvoie l'image des risques en noir et blanc.

```
def image_risques(releve):
   im = image_gris(releve)
   for i in range(len(im)):
      for j in range(len(im[0])):
         if im[i][j]<0.75:
            im[i][j]=0
         else:
            im[i][j]=1
   return(im)
```