Lucie CHAMBRAGNE DS-9 - Info. MPSI 1

I-

1)
```
SELECT idpatient FROM MEDICAL
WHERE etat = 'hernie discale' ;
```

2)
```
SELECT nom, prenom FROM PATIENT
JOIN MEDICAL ON PATIENT.id = MEDICAL.idpatient
WHERE etat = 'spondylolisthésis' ;
```

3)
```
SELECT etat, COUNT(idpatient) FROM MEDICAL
GROUP BY etat ;
```

4)
L'intérêt est l'efficacité. ?

5)
On sait que N = 100 000
n = 6.
8 bits = 1 octet
32 bits = 4 octets.
Donc pour le tableau il y a 6 × 100 000 réels codés sur 4 octets.
↳ 100000 × 6 × 4 = 2,4 Mo. (pour le tableau).
Donc pour stocker le tableau, il faut une mémoire de 2,4 Mo.

Pour le vecteur :
on sait que sa taille est $N = 100\,000$ contenant des valeurs entières codées sur 8 bits = 1 octet.
Donc pour stocker le vecteur, il faut une mémoire de 100 000 octets c'est-à-dire $0,1$ Mo.

Ainsi la quantité de mémoire totale nécessaire pour stocker le tableau et le vecteur des données est de $2,5$ Mo.

6)

```
def separationParGroupe(data, etat):
    N0 = []
    N1 = []
    N2 = []
    for i in range(len(data)):
        if etat[i] == 0:
            N0 = N0.append(data[i,:])
        elif etat[i] == 1:
            N1 = N1.append(data[i,:])
        else:
            N2 = N2.append(data[i,:])
    return [N0; N1; N2]
```

7)

ARGS1 = (n, n, i et j)
ARGS2 = (groupe[i], groupe[j], marker = mark[k])
ARGS3 = (data[i])
TEST = i != j

8) Les diagrammes de la diagonale répertorie le nombre de personnes ayant un angle d'incidence du bassin particulier. Ceux hors diagonale présentent angle d'orientation du bassin en fonction de l'angle d'incidence du bassin.

9)

$$x_{normj} = \frac{x_j - \min(X)}{\max(X) - \min(X)}$$

Si $x_j = \min(X)$ on retrouve bien $x_{normj} = 0$.
Si $x_j = \max(X)$ on retrouve $x_{normj} = 1$.

10)

```
def min_max(X):
    min = X[0]
    max = X[0]
    for i in X :
        if i < min :
            min = i
        elif i > max :
            max = i
    return min, max
```

11)

```
def distance(z, data):
    d = 0
    k = []
    for k in range(N+1):
        for i in range(len(z)):
            d = d + (z[i] - data[k][i])**2
```

```
        h = h.append(sqrt(d))
    return h.
```

12)

La partie 1 crée une liste T triée, de listes comprenant 2 éléments.

T représente une liste.
dist représente une liste.
select est aussi une liste.
ind est un entier.

15)

```
def moyenne(x):
    s = 0
    for t in x:
        s = s + t
    return s/len(x)

def variance(x):
    sc = 0
    for t in x:
        sc = sc + t**2
    return sc/len(x) - moyenne(x)**2.
```