

DS09

- Q1 select idpatient from MEDICAL where etat = "hernie discale"; ✓
- Q2 select nom, prenom from PATIENT join MEDICAL on patient.id = medical.
idpatient where etat = "spondylolisthésis"; ✓
à la suite
- Q3 select etat, count(idpatient) from MEDICAL group by etat. ✓
- Q4 utiliser la bibliothèque numpy car plus efficace car elle est plus rapide et utilise des données de même nature (entier, flottant).
Il faut être plus précis
- Q5 - 32 bits = 4 octets, $N = 100\,000$, $n = 6$ colonnes
pour data, il faut $N \times n \times 4 = 2,4$ Mo de mémoire.
- 8 bits = 1 octet
pour etat, il faut $N \times 1 \times 1 = 0,1$ Mo de mémoire.
Au total, il faut **2,5 Mo** de mémoire pour les deux tableaux. ✓
- Q6 def separationParGroupes(data, etat):
 groupe0 = []
 groupe1 = []


```

groupe2 = []
for i in range(len(data)):
    if etat[i] == 0:
        groupe0.append(data[i])
    elif etat[i] == 1:
        groupe1.append(data[i])
    elif etat[i] == 2:
        groupe2.append(data[i])
return groupe0, groupe1, groupe2.

```

Il faut
privilégier les
elif

Q7 $k = i \times j$
le tableau est de taille $n \times n$
avec un décalage d'indice :
ARGS 1 = (n, n, (i+1) * (j+1))

$data_x = groupes[k][i]$, $data_y = groupes[k][j]$, $marker = mark[k]$
ARGS 2 = groupes[k][i], groupes[k][j], marker = mark[k]

il faut utiliser data pour le histogramme


ARGS 3 = data[i]

[i, j]

pour savoir si en est ou pas sur la diagonale

TEST = i != j



Q8 diagrammes de la diagonale : pour voir si les valeurs sont
réparties autour d'une moyenne. 
diagrammes hors diagonale : ...

Q9 $z_{normj} = \frac{z_j - \min(X)}{\max(X) - \min(X)}$ ✓

Q10

```

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

```

Justification de la complexité ?

Q11

```

def distance(z, data):
    d = []
    for i in range(len(data)):
        so = 0
        for j in range(len(data[i])):
            so += (z[j] - data[i][j])**2
        d.append(sqrt(so))
    return d

```

Q12

partie 1: on crée un tableau où on stocke des distance min.

partie 2:

partie 3:

T: liste qui contient les distances

dist: valeur des distances (Q11)

select:

ind:

Liste triée

Q13

Q14

Q15

def moyenne(X):

s = 0

for i in X:

s += i

return s / len(X)



def variance(X):

s = 0

m = moyenne(X)

for i in X:

s += (i - m) ** 2

return s / len(X)



Q16

def synthese(data, etat):

Q17

def gaussienne(a, moy, v):

Q18

def probabilite_groupe(z, data, etat):

Q19

def prediction(...):

Q20

Q21