CHEVALIER
Valentin

Informatique

1) SELECT idpatient FROM MEDICAL WHERE état = 'hernie discale' ✓

2) SELECT nom, prenom FROM PATIENT JOIN MEDICAL ON PATIENT.id = idpatient WHERE état = 'spondylolisthésis' ✓

3) SELECT état, COUNT(idpatient) as nb FROM MEDICAL
GROUP BY état

Distinct

4) Le module Array de numpy et sa représentation matricielle rendent les tableaux de grande taille plus lisibles ✓

5) On a 6 colonnes et 100000 lignes pour un total de 600 000 valeurs en chacune est un réel de 32 bits soit 32 x 600000 bits au total.
Dans le vecteur on a 100000 entiers codés sur 8 bits
Pour tout stocker il faut donc 600000 x 32 x 8 x 100000 = 80 000 000 octets
Soit 2,5 Mo. ✓

6)
```
def separationParGroupe(data, etat):
    liste = [[],[],[]]
    for i in range(len(etat)):
        if etat[i][0]==0:
            liste[0].append(data[i])
        elif etat[i][0]==1:
            liste[1].append(data[i])
        elif etat[i][0]==2:
            liste[2].append(data[i])
    return liste
```

7) ARGS1 = m, n, i*m+j+1

8) $x_{norm_j} = \frac{x_j - \min(x)}{\max(x) - \min(x)}$

10)
```
def min_max(X):
    min = X[0]
    max = 0
    for i in range(len(X)):
        if X[i][0] = min:
            min = X[:][0]
        elif X[:][0] = max:
            max = X[:][0]
    return(min, max)
```

Un mot sur la complexité ?

11)
```
def distance(g, data):
    tab = data.copy()
    for i in range(len(data[0])):
        col = []
        for j in range(len(data)):
            col.append(data[j][i])
        masc, minn = max_min(col)
        for j in range(len(data)):
            data[j][i] = (data[j][i] - minn)/(masc - minn)
    g[i] = (g[i] - minn)/(masc - minn)
    for j in range(len(data)):
        tab[j][i] = abs(data[j][i] - g[i])
    return tab
```

Il faut prendre une autre norme

12) La partie 1 crée la liste T à partir des distances puis la trie

✓

14) On remarque que la courbe est toujours au dessus des 70%
donc l'algorithme est assez efficace mais peut quand même faire des erreurs.

✓