

DS d'Informatique

Q1)

```
SELECT id_patient  
FROM MEDICAL  
WHERE etat = 'hernie discale'
```

Q2)

```
SELECT nom, prenom  
FROM PATIENT  
WHERE id IN (SELECT id_patient  
FROM MEDICAL  
WHERE etat = 'spondylolisthésis')
```

Q3)

```
SELECT COUNT(*)  
FROM MEDICAL  
GROUP BY etat
```

Q4) Numpy est plus rapide.

Q5)

Q6)

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

Q7) ARG1: $n, n, n(i-1) + j$
 ARG2: $groupes[k][i], groupes[k][j], marker = mark[k]$
 ARG3: $data[i]$
 TEST: $i \neq j$

Q8) Les diagrammes diagonaux permettent de mesurer le taux d'apparition des symptômes.
 Les diagrammes hors diagonale permettent de chercher un lien entre les symptômes selon la maladie observée.

Q9)
$$x_{normj} = \frac{x_j}{max(x) - min(x)}$$

Q10) def min_max(x):
 min = x[0]
 max = x[0]
 for i in range (len(x)):
 if x[i] < min:


```

min = x[i]
if x[i] > max:
    max = x[i]
return max, min

```

Q11)

```

def distance(x, data):
    list = []
    for i in range(len(x)):
        x = 0

```

Q12) La partie 1 crée une liste T contenant la distance et l'indice correspondant à chaque patient, triée par distances. La partie 2 crée une liste select qui donne la distance la plus courte pour chaque symptôme. La partie 3 cherche le maximum de select pour donner l'état correspondant, stocké dans ind.

Q14) L'algorithme ne réussit jamais à plus de 75%, il n'est pas très efficace.

Q15)

```

def moyenne(x):
    n = len(x)
    a = 0
    for i in range(n):
        a = a + x[i]
    return a / n

```

```

def variance(x):
    m = moyenne(x)
    n = len(x)
    a = 0
    for i in range(n):
        a = a + (x[i] - m) ** 2
    return a / n

```

Q16)

```

def synthese(data, etat):
    l1 = len(data)
    l2 = len(etat)
    tab = []
    for i in range(l2):
        tab.append([])
    for k in range(l1):
        tab[etat[k]].append([moyenne(data[k]), variance(data[k])])
    return tab

```