BROCARD
Laureen

MPSI 1
18/06/2020

DS n° 9 : info

Question 1 :

```
SELECT idpatient
FROM MEDICAL
WHERE etat = 'hernie viscale' ;
```

Question 2 :

```
SELECT A.nom , A.prenom
FROM PATIENT AS A
JOIN MEDICAL AS B
ON A.id = B.idpatient
WHERE B.etat = 'spondylolisthesis' ;
```

Question 3

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

Question 4 :

Les opérations sur les tableaux numpy sont plus rapides qu'avec des listes.

L'accès à un élément d'un tableau numpy se fait en temps constant. Il n'y a pas besoin de parcourir tout le tableau, ce qui s'avère très pratique quand le tableau est très grand

Question 5 :

Le tableau contient $N = 100\,000$ lignes et $n = 6$ colonnes
Le vecteur est de taille $N = 100\,000$

Le tableau peut contenir $N \times n = 600\,000$ valeurs
Le vecteur peut contenir $N = 100\,000$ valeurs

En norme IEEE 754 chaque nombre est représenté sur 64 bits

Il faut donc $64 \times 600\,000 = 38\,400\,000$ bits pour stocker le tableau et $64 \times 100\,000 = 6\,400\,000$ bits pour stocker le vecteur, soit $44\,800\,000$ bits au total

Or 1 octet = 8 bits
Donc $44\,800\,000$ bits $= \frac{44\,800\,000}{8}$ octets

i.e. 5 600 000 octets
i.e. 5,6 Mo

Question 6

def separationParGroupe(data, etat):

Question 7

ARGS1 = N, n, j

ARGS2 = incidence_bassin,

TEST = if i != j

~~Question~~ 8

~~Les diagrammes de la diagonale~~

Question 12

- Ligne 3 : on crée une liste vide T
  Ligne 4 : on range dist, la liste des distances entre z et chaque élement de data
  Ligne 5, 6 : Pour chaque valeur de dist, on ajoute à T la liste composée d'un élement de dist et la valeur de l'état
  Ligne 7 : On trie T

↳

Question 14 :

La courbe obtenue est irrégulière bien que des valeurs en ordonné soient environ comprises entre 70% et 75%
Le pourcentage de réussite n'est pas proportionnelle au nombre de voisin

Question 15

```
def moyenne(x):
    mu = 0
    for i in range(n):
        mu = mu + x[i]
    return 1/n * mu
```

~~def variance(x):~~
~~sigma_carre = 0~~
~~for i in range(n):~~
~~sigma_carre = x[i] - nu~~

def variance(x):
    sigma_carre = 0
    for i in range(n):
        sigma_carre = (x[i] - nu) ** 2
    return $\frac{1}{n}$ x sigma_carre