TD1 Python

Exercice 1 (Bases)

In [ ]:
```python
a = 7 # Affectation de variables
print(a) # Afficher le contenu d'une variable

b = a + 4 # Addition
c = a*a # Multiplication
d = a**2 # Puissance

# Booléens et comparaison

print(c == d) # Pour vérifier l'égalité entre deux variables
print(a < b)
print(c >= b)
```

1. Commentez les sorties de R suite aux intructions suivantes.

In [ ]:
```python
import numpy as np # Important d'une librairie

# print(np.log(-1))
# print(1/0)
# print(0/0)
# print(np.log(2))
```

Exercice 2 (Affectation)

1. Creation d'un array dont les valeurs sont les suivantes 10, 5, 3, 6, 21. Utiliser la fonction `np.array`.

In [ ]:

2. Extraire successivement le deuxième et le troisème élément de a.

In [ ]:

3. Faire un array semblable à une matrice de taille 5×1, pour cela, utilisez la méthode `.reshape` sur l'array.

In [ ]:

4. Pour voir la différence entre un array vecteur et un array matrice, regarder ses dimensions avec la méthode `.shape`.

In [ ]:

5. Utiliser la fonction `np.arange` pour pour définir un vecteur d composé des 5 premiers entiers impairs.

In [ ]:

6. Créer un vecteur `e` compoé des 5 premiers entiers.

In [ ]:

7. A l'aide de la fonction `np.repeat`, générer l'array `(1, 1, 2, 2, 3, 3)`.

In [ ]:

8. Créer la matrice diagonale de dimension 3 dont les éléments de la diagonale sont 1, 5 et 9. Tips: utilisez `np.diag`

In [ ]:

Exercice 3 (Manipulation de base)

1. Taper `2*a + b + 1, e[3], cos(a), exp(a)`. Qu'obtient-on ?

In [ ]:

2. Taper `m_1*m_2` puis `m_1@m_2`. Comparer et expliquer quelles sont les deux opérations effectuées.

In [ ]:
```python
m_1, m_2 = np.ones((3,3)), 2*np.ones((3,3))
```

3. Taper `v_1@v_2`. Que se passe-t-il? Corriger par `v_1@v_2.T`.

In [ ]:
```python
# c. Transposée
v_1 = np.array([1,2,3]).reshape((-1, 1))
v_2 = np.array([1,2,3]).reshape((-1, 1))
```

4. Concatenez les array `v_1` et `v_2` de manière verticale ou horizontale en utilisant la function `np.concatenate`

In [ ]:

5. Utilisez les méthodes `.sum`, `.mean`, `.std`. Quels sont les résultats ?

In [ ]:

6. Opérations sur les arrays de booléens.

In [ ]:
```python
array_bool = np.array([True, True, False, True])
array_bool.all()
array_bool.any()
```

Exercice 4 (Matrices)

1. Créer la matrice `mat` en execuctant les instructions suivantes.

In [ ]:
```python
mat = np.array([1, 0, 3, 4, 5, 5, 0, 4, 5, 6, 3, 4, 0, 1, 3, 2]).reshape((4, 4))
mat
```

2. Créer un array dont les éléments sont ceux de la diagonale de `mat`.

In [ ]:

3. Créer la matrice `submat` contenant les deux premières lignes de `mat`, et la matrice `submat2` contenant les deux dernières colonnes de `mat`.

In [ ]:

4. Créer la matrice `smallmat` composée des colonnes de `mat` dont tous les éléments sont inférieurs à 5 (utiliser apply et all), et la matrice `notzero` composée des lignes de `mat` dont tous les éléments sont non nuls.

In [ ]:

Exercice 5 (Résolution d'un système linéaire)

Soit

$$A = \begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 2 \\ 0 & 3 & 1 \end{pmatrix}, b = \begin{pmatrix} 1 \\ 2 \\ 1 \end{pmatrix}$$

Nous allons résoudre le système $Ax = b$

1. Définir la matrice A et le vecteur b

In [ ]:

2. La matrice A est-elle inversible ? Calculer son déterminant avec `np.linalg.det`

In [ ]:

3. Résoudre le système linéaire Ax=b en utilisant la fonction `np.linalg.solve`

In [ ]:

Exercice 6 (Boucle for)

1. Créer une boucle `for` qui permet de calculer la factorielle d'un nombre.

In [ ]:

2. Réutiliser le code de la question précédente pour en faire une fonction qui prend comme argument un entier positif, et qui retourne sa factorielle. Avant d'effectuer le calcul, la fonction vérifie le type de l'argument. Si celui-ci n'est pas conforme, alors la fonction retourne un message d'erreur.

In [ ]:

Exercice 7 (Divisibilité)

Ecrire une fonction `r_function` qui prend deux arguments k et q. Cette fonction vérifie si K et Q sont des entiers positifs. Ensuite, elle considère le vecteur des entiers de 1 à K, et détermine combien d'éléments de ce vecteur sont exactement divisible par Q.

In [ ]:

Exercice 8 (Fibonacci)

Utiliser une boucle `for` pour reproduire la suite de Fibonacci jusqu'à son dixième terme (la séquence F_n est définie par la relation de récurrence suivante : $F_n = F_{n-1} + F_{n-2}$; les valeurs initiales sont $F_0 = 0$ et $F_1 = 1$).

In [ ]:

Exercice 9 (Racine d'un polynôme et graphiques)

$y = 2x^2 - 8x + 6$

En rappelant la formule du discriminant, calculer les racines avec `python`. Puis tracer sur un graphique à l'aide de la librairie `matplotlib` et de son module `pyplot` la courbe représentative du polynôme du second degré sur l'intervalle $[0, 5]$. Tracer la droite horizontale d'équation $y = 0$, les droites verticales passant par les racines en les faisant figurer sur le graphe (e.g., avec la fonction `axvline`).

In [ ]:
```python
import matplotlib.pyplot as plt
```

Exercice 10 (Graphiques suite)

1. Construire une fonction qui calcule la valeur de la fonction f définie par $f(x) = \sin(x)^2 + \sqrt{|x-3|}$.

In [ ]:

2. De même avec la fonction suivante

$$g(x) = \begin{cases} \sin(x)\ln(x) & x > 0 \\ 3 + \sin^2(x) & x \leq 0. \end{cases}$$

In [ ]:

3. Tracer la courbe représentative des deux fonctions sur le domaine $[-6, 3]$.

```
In [ ]:
```

TP 1: Prise en main du logiciel R

fabien.navarro@univ-paris1.fr

2024-01-24

Exercice 1 (Introduction à `R`)

1. Commentez les sorties de R suite aux intructions suivantes.

- `log(-1)`
- `1/0`
- `0/0`
- `log(2, base=23)`
- `round(exp(1),digit=3)`
- `asin(3/11)`

2. Affectation

(a) En utilisant la fonction `c`, affecter les valeurs $(10, 5, 3, 6, 21)$ au vecteur (au sens de R) a. Afficher a.

(b) Extraire successivement le deuxième et le troisème élément de a.

(c) En utilisant la fonction `matrix`, affecter les valeurs $(15, 3, 12, 2, 1)$ au vecteur (au sens mathématique) b. Retourner b.

(d) Pour voir la différence entre un vecteur et une matrice. Demander `nrow(a), ncol(a), dim(a)` et de même pour b.

> Dans toute la suite, on considère des vecteurs au sens de R (i.e., sans dimension).

(e) Utiliser la fonction `seq` pour définir un vecteur d composé des 5 premiers entiers impairs.

(f) Créer un vecteur e compoé des 5 premiers entiers.

(g) A l'aide de la fonction `rep`, générer les vecteurs: `(1, 2, 3, 1, 2, 3)` et `(1, 1, 2, 2, 3, 3)`.

(h) Créer la matrice diagonale de dimension 3 dont les éléments de la diagonale sont 1, 5 et 9.

3. Manipulation de base

(a) Taper `2*a + b + 1, e[3], cos(a), exp(a)`. Qu'obtient-on ?

(b) Taper `a*e` puis `a%*%e`. Comparer.

(c) Taper `b%*%e`. Que se passe-t-il? Corriger par `f <- t(b)%*%e`.

(d) Taper `cbind(b, e)` puis `rbind(b, e)`.

(e) Taper `c(a,b)`.

(f) Taper `sum(a), range(a), length(a) et summary(a)`

(g) Saisir le vecteur u en tapant `u <- c(TRUE, FALSE, TRUE, TRUE)`. Evaluer les instructions suivantes: `!u, u | !u, any(u) et all(u)`.

Exercice 2 (Manipulation de matrices)

1. Créer la matrice `mat` en executant les instructions suivantes.

```
values <- c(1, 0, 3, 4, 5, 5, 0, 4, 5, 6, 3, 4, 0, 1, 3, 2)
mat <- matrix(values, nrow = 4, ncol = 4)
mat
```

```
##      [,1] [,2] [,3] [,4]
## [1,]    1    5    5    0
## [2,]    0    5    6    1
## [3,]    3    0    3    3
## [4,]    4    4    4    2
```

2. Créer le vecteur `vec` contenant les éléments de la diagonale de `mat`.

3. Créer la matrice `submat` contenant les deux premières lignes de `mat`, et la matrice `submat2` contenant les deux dernières colonnes de `mat`.

4. Créeer la matrice `smallmat` composée des colonnes de `mat` dont tous les éléments sont inférieurs à 5 (utiliser apply et all), et la matrice `notzero` composée des lignes de `mat` dont tous les éléments sont non nuls.

5. Générer la matrice `mat2` par la commande suivante `mat2 <- matrix(1:16, nrow=4)`, puis saisir et comparer les deux commandes suivantes: `mat * mat2` et `mat % * % mat2`.

Exercice 3 (Résolution d'un système linéaire)

Soient

$$A = \begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 2 \\ 0 & 3 & 1 \end{pmatrix}, \quad b = \begin{pmatrix} 1 \\ 2 \\ 1 \end{pmatrix}.$$

1. Définir la matrice A et le veteur b.

```
A <- matrix(c(1, 0, 0, 0, 1, 3, 0, 2, 1), nrow = 3, ncol = 3)
b <- c(1, 2, 1)
```

2. La matrice A est-elle inversible ? Calculer son déterminant.

3. Résoudre le système linéaire $Ax = b$ (en utilisant la fonction `solve`).

Exercice 4 (Boucle for)

1. Créer une boucle `for` qui permet de calculer la factorielle d'un nombre.

2. Réutiliser le code de la question précédente pour en faire une fonction qui prend comme argument un entier positif, et qui retourne sa factorielle. Avant d'effectuer le calcul, la fonction vérifie le type de l'argument. Si celui-ci n'est pas conforme, alors la fonction retourne un message d'erreur.

Exercice 5 (Divisibilité)

Ecrire une fonction `R` qui prend deux arguments K et Q. Cette fonction vérifie si K et Q sont des entiers positifs. Ensuite, elle considère le vecteur des entiers de 1 à K, et détermine combien d'éléments de ce vecteur sont exactement divisible par Q.

Exercice 6 (Fibonacci)

Utiliser une boucle `for` pour reproduire la suite de Fibonacci jusqu'à son dixième terme (la séquence F_n est définie par la relation de récurrence suivante : $F_n = F_{n-1} + F_{n-2}$; les valeurs initiales sont $F_0 = 0$ et $F_1 = 1$).

Exercice 7 (Racine d'un polynôme et graphiques)

On souhaite trouver les solutions de l'équation

$$2x^2 - 8x + 6 = 0$$

En rappelant la formule du discriminant, calculer les racines avec `R`. Puis tracer sur un graphique à l'aide de la fonction `curve` la courbe représentative du polynôme du second degré sur l'intervalle $[0, 5]$. Tracer la droite horizontale d'équation $y = 0$, les droites verticales passant par les racines en les faisant figurer sur le graphe (e.g., avec la fonction `points`).

Exercice 8 (Graphiques suite)

1. Construire une fonction qui calcule la valeur de la fonction f définie par

$$f(x) = \sin(x)^2 + \sqrt{|x - 3|}.$$

2. Tracer la courbe représentative de la fonction f sur le domaine $[-6, 3]$.

3. Ajouter sur le graphique précédent la courbe correspondant à la fonction g définie par

$$g(x) = \begin{cases} \sin(x)\ln(x) & x > 0 \\ 3 + \sin^2(x) & x \leq 0. \end{cases}$$