

Info. - TD : Feuille 3

Exercice 1 – créations numpy

Dessiner et/ou expliquer le contenu (et la forme) du tableau créé par

Q1) `np.array([4, 5])`.

Q5) `np.random.normal(2, 3, (4, 5))`.

Q2) `np.zeros((4, 5))`.

Q6) `np.linspace(3, 4, 5)`.

Q3) `np.ones((5, 4))`.

Q4) `np.random.uniform(2, 3, (4, 5))`.

Q7) `np.arange(100, 1000, 10)`.

Q8) Donner 3 façons de créer un tableau numpy suivants (pas forcément des entiers) ?

1 `[100 125 150 175 200]` # parfois noté: `array([100, 125, 150, 175, 200])`

Exercice 2 – *shape* numpy

Q9) Étant donné une variable `a` contenant un tableau numpy, comment afficher sa *shape* (sa forme, ses dimensions) ?

.

Q10) Quelle est la *shape* (la forme, les dimensions) du tableau créé par `np.array([4, 5])` ?

Q11) Quelle est la *shape* du tableau créé par `np.zeros((4, 5))` ?

Q12) Quelle est la *shape* du tableau créé par `np.ones((5, 4))` ?

Q13) Quelle est la *shape* du tableau créé par `np.random.uniform(2, 3, (4, 5))` ?

Q14) Quelle est la *shape* du tableau créé par `np.random.normal(2, 3, (4, 5))` ?

Q15) Quelle est la *shape* du tableau créé par `np.linspace(10, 20, 30)` ?

Q16) Quelle est la *shape* du tableau créé par `np.arange(100, 1000, 10)` ?

.

Q17) Quelle est la *shape* du tableau créé par `np.ones((100,))` ?

Q18) Quelle est la différence entre `np.zeros((100,))`, `np.zeros((100, 1))` et `np.zeros((1, 100))` ?

.

Q19) Quelle est la *shape* du tableau créé par `np.ones((100)).reshape(( 5, 20))` ?

Q20) Quelle est la *shape* du tableau créé par `np.ones((100)).reshape((-1, 20))` ?

Q21) Quelle est la *shape* du tableau créé par `np.ones((100)).reshape(( 5, -1))` ?

Q22) Quelle est la *shape* du tableau créé par `np.ones((100)).reshape((-1, 5))` ?

.

Q23) Dessiner et/ou expliquer le tableau créé par `np.linspace(2, 5, 16).reshape((2, 8))`.

Q24) Dessiner et/ou expliquer le tableau créé par `np.linspace(2, 5, 16).reshape((-1, 4))`.

Exercice 3 – opérations

Q25) Quel code Python (concis) permet de créer le tableau numpy suivant, que l'on appellera **a** (et que l'on pourra donc réutiliser par la suite) ?

```
1 array([100, 110, 120, 130, 140, 150, 160, 170, 180, 190, 200, 210, 220,
2       230, 240, 250, 260, 270, 280, 290, 300, 310, 320, 330, 340, 350,
3       360, 370, 380, 390, 400, 410, 420, 430, 440, 450, 460, 470, 480,
4       490, 500, 510, 520, 530, 540, 550, 560, 570, 580, 590, 600, 610,
5       620, 630, 640, 650, 660, 670, 680, 690, 700, 710, 720, 730, 740,
6       750, 760, 770, 780, 790, 800, 810, 820, 830, 840, 850, 860, 870,
7       880, 890, 900, 910, 920, 930, 940, 950, 960, 970, 980, 990])
```

Q26) En notant bien la différence avec la question précédente, quel code Python (concis) permet de créer le tableau numpy suivant, que l'on appellera **b** ?

```
1 array([[100, 110, 120, 130, 140, 150, 160, 170, 180, 190],
2       [200, 210, 220, 230, 240, 250, 260, 270, 280, 290],
3       [300, 310, 320, 330, 340, 350, 360, 370, 380, 390],
4       [400, 410, 420, 430, 440, 450, 460, 470, 480, 490],
5       [500, 510, 520, 530, 540, 550, 560, 570, 580, 590],
6       [600, 610, 620, 630, 640, 650, 660, 670, 680, 690],
7       [700, 710, 720, 730, 740, 750, 760, 770, 780, 790],
8       [800, 810, 820, 830, 840, 850, 860, 870, 880, 890],
9       [900, 910, 920, 930, 940, 950, 960, 970, 980, 990]])
```

Q27) Quel code Python (concis) permet de créer le tableau numpy suivant, que l'on appellera **c** ?

```
1 array([[ 1.,  1.,  1.,  1.,  1.,  1.,  1.,  1.,  1.,  1.],
2       [ 1.,  1.,  1.,  1.,  1.,  1.,  1.,  1.,  1.,  1.],
3       [ 1.,  1.,  1.,  1.,  1.,  1.,  1.,  1.,  1.,  1.],
4       [ 1.,  1.,  1.,  1.,  1.,  1.,  1.,  1.,  1.,  1.],
5       [ 1.,  1.,  1.,  1.,  1.,  1.,  1.,  1.,  1.,  1.],
6       [ 1.,  1.,  1.,  1.,  1.,  1.,  1.,  1.,  1.,  1.],
7       [ 1.,  1.,  1.,  1.,  1.,  1.,  1.,  1.,  1.,  1.],
8       [ 1.,  1.,  1.,  1.,  1.,  1.,  1.,  1.,  1.,  1.],
9       [ 1.,  1.,  1.,  1.,  1.,  1.,  1.,  1.,  1.,  1.]])
```

Q28) Quel code Python (concis) permet de créer le tableau numpy suivant, que l'on appellera **d** ?

```
1 array([[ 1000.,  1000.,  1000.,  1000.,  1000.,  1000.,  1000.,  1000.,  1000.,  1000.],
2       [ 1000.,  1000.,  1000.,  1000.,  1000.,  1000.,  1000.,  1000.,  1000.,  1000.],
3       [ 1000.,  1000.,  1000.,  1000.,  1000.,  1000.,  1000.,  1000.,  1000.,  1000.],
4       [ 1000.,  1000.,  1000.,  1000.,  1000.,  1000.,  1000.,  1000.,  1000.,  1000.],
5       [ 1000.,  1000.,  1000.,  1000.,  1000.,  1000.,  1000.,  1000.,  1000.,  1000.],
6       [ 1000.,  1000.,  1000.,  1000.,  1000.,  1000.,  1000.,  1000.,  1000.,  1000.],
7       [ 1000.,  1000.,  1000.,  1000.,  1000.,  1000.,  1000.,  1000.,  1000.,  1000.],
8       [ 1000.,  1000.,  1000.,  1000.,  1000.,  1000.,  1000.,  1000.,  1000.,  1000.],
9       [ 1000.,  1000.,  1000.,  1000.,  1000.,  1000.,  1000.,  1000.,  1000.,  1000.]])
```

Q29) Quel code Python (concis) permet de créer le tableau numpy suivant, que l'on appellera **e** ?

```
1 array([[ 1100.,  1110.,  1120.,  1130.,  1140.,  1150.,  1160.,  1170.,  1180.,  1190.],
2       [ 1200.,  1210.,  1220.,  1230.,  1240.,  1250.,  1260.,  1270.,  1280.,  1290.],
3       [ 1300.,  1310.,  1320.,  1330.,  1340.,  1350.,  1360.,  1370.,  1380.,  1390.],
4       [ 1400.,  1410.,  1420.,  1430.,  1440.,  1450.,  1460.,  1470.,  1480.,  1490.],
5       [ 1500.,  1510.,  1520.,  1530.,  1540.,  1550.,  1560.,  1570.,  1580.,  1590.],
6       [ 1600.,  1610.,  1620.,  1630.,  1640.,  1650.,  1660.,  1670.,  1680.,  1690.]])
```

```

7         [ 1700., 1710., 1720., 1730., 1740., 1750., 1760., 1770., 1780., 1790.],
8         [ 1800., 1810., 1820., 1830., 1840., 1850., 1860., 1870., 1880., 1890.],
9         [ 1900., 1910., 1920., 1930., 1940., 1950., 1960., 1970., 1980., 1990.]]))

```

Q30) Donner 2 façons de créer la variable `e` directement (sans utiliser les variables précédentes).

.

Q31) Donner le code Python (sans `np.array`) qui permet de créer le tableau numpy suivant :

```

1 array([[ 10,  40,  90, 160, 250],
2        [ 360, 490, 640, 810, 1000]])

```

Exercice 4 – indices numpy

On considère le tableau `e` de l'exercice précédent, mais vous devez donner des réponses les plus générales possibles. Par exemple, on évitera de supposer que `e` a 9 lignes et 10 colonnes.

Comment...

Q32) ... extraire la première valeur de la première ligne ?

Q33) ... remplacer la première valeur de la première ligne par 999 ?

Q34) ... extraire la première valeur de la seconde ligne ?

Q35) ... remplacer la première valeur de la seconde ligne par 999 ?

Q36) ... extraire la première valeur de la troisième ligne ?

Q37) ... extraire la première valeur de la ligne d'indice 6 ?

Q38) ... extraire la dernière valeur de la ligne d'indice 3 ?

Q39) ... remplacer la dernière valeur de la ligne d'indice 3 par 999 ?

Q40) ... extraire l'avant-avant-dernière valeur de la ligne d'indice 3 ?

Q41) ... remplacer l'avant-avant-dernière valeur de la ligne d'indice 3 par 999 ?

Q42) ... extraire l'avant-dernière valeur de la dernière ligne ?

Q43) ... remplacer l'avant-dernière valeur de la dernière ligne par 999 ?

Exercice 5 – tranches numpy

Toujours à partir de `e` et en donnant des réponses générales, comment ...

Q44) ... extraire la première colonne du tableau ?

Q45) ... remplacer toutes les valeurs de la première colonne du tableau par 999 ?

Q46) ... extraire la seconde colonne du tableau ?

Q47) ... remplacer toutes les valeurs de la seconde colonne du tableau par 999 ?

Q48) ... remplacer toutes les valeurs de la seconde colonne du tableau par les nombres de 1 à 9 ?

Q49) ... extraire la colonne d'indice 6 ?

Q50) ... extraire la ligne d'indice 6 ?

Q51) ... extraire les valeurs d'indices 4 à 7 (inclus) de la seconde ligne du tableau ?

Q52) ... remplacer toutes les valeurs d'indices 4 à 7 (inclus) de la seconde ligne du tableau par 999 ?

Q53) ... remplacer toutes les valeurs d'indices 4 à 7 (inclus) de la seconde ligne du tableau par les nombres de 24 à 27 ?

- Q54)** ... remplacer toutes les valeurs d'indices 4 à 7 (inclus) de la seconde ligne du tableau par les puissances de 10, c'est à dire les nombres 10, 100, 1000, 1000 ?
- Q55)** ... remplacer toutes les valeurs d'indices 4 à 7 (inclus) de la seconde ligne du tableau par celles de la troisième ligne ?
- Q56)** ... remplacer toutes les valeurs d'indices 4 à 7 (inclus) de la seconde ligne du tableau par le carré de leurs valeurs ?
- Q57)** ... extraire les 4 premières valeurs de la seconde ligne du tableau ?
- Q58)** ... extraire les 4 dernières valeurs de la seconde ligne du tableau ?
- Q59)** ... extraire les 4 dernières valeurs de la dernière ligne du tableau ?
- Q60)** ... remplacer les 4 dernières valeurs de la dernière ligne du tableau par le carré de leurs valeurs ?
- Q61)** ... extraire une valeur sur deux de la colonne d'indice 3 ?
- Q62)** ... remplacer une valeur sur deux de la colonne d'indice 3, par la valeur 999 ?
- Q63)** ... remplacer une valeur sur deux de la colonne d'indice 3, par les valeurs de 0 à N-1 (s'il y a N éléments à remplacer) ?
- Q64)** ... extraire une valeur sur deux mais prises de bas en haut de la colonne d'indice 3 ?
- Q65)** ... extraire une valeur sur deux des colonnes de l'indice 3 à l'avant-avant-dernière (incluses) ?
- Q66)** ... remplacer une valeur sur deux des colonnes de l'indice 3 à l'avant-avant-dernière (incluses), par la valeur 999 ?
- Q67)** ... remplacer une valeur sur deux des colonnes de l'indice 3 à l'avant-avant-dernière (incluses), par les valeurs de 0 à N-1 (s'il y a N éléments à remplacer) ?
- Q68)** ... extraire une valeur sur deux des colonnes d'indice 3 à 5 (inclus) ?
- Q69)** ... extraire une valeur sur deux des colonnes de l'indice 3 à la dernière (incluses) ?
- Q70)** ... extraire une valeur sur deux des colonnes de l'indice 3 à l'avant-avant-dernière ?
- Q71)** ... extraire une valeur sur deux, en commençant à l'indice 3, des colonnes de l'indice 3 à la dernière (incluses) ?
- Q72)** Que fait `e[:, -2, :: -2]` ?
- Q73)** et `e[:, 2, :: 2] = e[:, -2, :: -2]` ?
- Q74)** et `e[-1:2:-3, -1:2:-3]` ?
- Q75)** et `e[-2:2:-2, -2:2:-2]` ?

Rappels : extraction et modification de tableau (et liste)

Si `t` est un tableau numpy (ou une liste python), il est possible d'extraire une sous-partie avec `t[???`]. Si `t` est de dimension 1, alors `???` peut être soit un indice simple soit une tranche. D'une manière générale (si `t` a plusieurs de dimensions), `???` doit être un ensemble d'indices (ou de tranches) séparés par des virgules, par exemple de la forme `t[???,???,???,???` si `t` a 4 dimensions.

Le même genre de choses peuvent être placées à gauche d'un `=` pour changer la ou les valeurs dans le tableau, par exemple `t[???] = 4`, remplace la ou les valeurs de la sous-partie de `t`, chacune par la valeur 4. Il est aussi possible de mettre une liste ou un tableau à droite du `=` de façon à remplacer les valeurs une à une,. **Exemple:** `t[-1,-2:] = [123, 456]` remplace, dans la dernière ligne de `t`, l'avant dernière valeur par 123 et la dernière valeur par 456.