

Page	Possible	Score
1	10	
2	10	
Total:	20	

UE12P26 – PE

Python numérique - Quiz du cours #5

20 octobre 2026

Cochez une seule réponse par question.

Ce quiz porte sur la séance 5 (le temps et les figures). Les questions 1 à 12 sont accessibles, les questions 13 à 20 sont plus techniques : lisez bien le code avant de répondre. Vous disposez de 10 minutes pour répondre aux 20 questions.

1. `df` a trois colonnes numériques. Que trace `df.plot()` ?

- ☐ Une courbe par colonne, avec l'index en abscisse et une légende automatique
- ☐ Une seule courbe : celle de la première colonne
- ☐ Un nuage de points de la première colonne en fonction de la seconde
- ☐ Une erreur : il faut préciser `x=` et `y=`

1 ☐

6. Quelle est la différence entre `plt.plot(x, y)` et `plt.scatter(x, y)` ?

- ☐ `plot` relie les points par des segments, `scatter` affiche des points isolés
- ☐ `plot` trace en 2D, `scatter` en 3D
- ☐ `scatter` trie les points selon `x`, pas `plot`
- ☐ Aucune : `scatter` est un alias de `plot`

6 ☐

2. Que fait `pd.to_datetime(df["date"], format="%Y-%m-%d")` ?

- ☐ Il formate les dates en chaînes de la forme `20/10/2026`
- ☐ Il transforme l'index de `df` en `DatetimeIndex`
- ☐ Il convertit la colonne de chaînes (ex. `"2026-10-20"`) en une `Series` de type `datetime64`
- ☐ Il extrait l'année, le mois et le jour dans trois colonnes

2 ☐

7. Quel code affiche deux courbes avec une légende « sin » / « cos » ?

- ☐ `plt.plot(x, y1, name="sin")`
`plt.plot(x, y2, name="cos")`
`plt.legend()`
- ☐ `plt.plot(x, y1, label="sin")`
`plt.plot(x, y2, label="cos")`
`plt.legend()`
- ☐ `plt.plot(x, y1, label="sin")`
`plt.plot(x, y2, label="cos")`
- ☐ `plt.plot(x, y1)`
`plt.plot(x, y2)`
`plt.title("sin, cos")`

7 ☐

3. `df["date"]` est de type `datetime64`. Quelle expression extrait l'année de chaque date ?

- ☐ `df["date"].year`
- ☐ `df["date"].dt.year`
- ☐ `df["date"].dt.year()`
- ☐ `df["date"].str[:4]`

3 ☐

8. Après `fig, axes = plt.subplots(2, 3)`, comment tracer une courbe dans la case en haut à droite ?

- ☐ `axes[2, 0].plot(x, y)`
- ☐ `axes[1, 3].plot(x, y)`
- ☐ `fig.plot(x, y, pos=3)`
- ☐ `axes[0, 2].plot(x, y)`

8 ☐

4. Que faut-il pour pouvoir appeler `df.resample("W").mean()` ?

- ☐ Que les dates soient triées par ordre décroissant
- ☐ Que la colonne de dates soit de type chaîne de caractères
- ☐ Rien : `resample` fonctionne sur n'importe quel index
- ☐ Que l'index de `df` soit un `DatetimeIndex` (par ex. après `df.set_index("date")`)

4 ☐

9. Comment donner un titre à la sous-figure `ax` (un objet `Axes`) ?

- ☐ `ax.set_title("Titre")`
- ☐ `ax.title("Titre")`
- ☐ `plt.title(ax, "Titre")`
- ☐ `fig.set_title("Titre")`

9 ☐

5. Quelle est la différence entre `s.resample("7D").mean()` et `s.rolling(7).mean()` ?

- ☐ Aucune : les deux calculent une moyenne sur 7 jours
- ☐ `rolling` regroupe par semaine calendaire, `resample` décale la série de 7 lignes
- ☐ `resample` découpe le temps en corbeilles disjointes (une valeur par corbeille) ; `rolling` utilise une fenêtre glissante (une valeur par point)
- ☐ `resample` ne fonctionne que sur des `Series`, `rolling` que sur des `DataFrame`

5 ☐

10. Dans un notebook, où placer `plt.savefig("fig.png")` pour enregistrer la figure que l'on vient de tracer ?

- ☐ Avant le premier `plt.plot`
- ☐ Après `plt.show()`
- ☐ Après les tracés et avant `plt.show()`
- ☐ Peu importe, la figure reste en mémoire

10 ☐

11. `df` contient les colonnes `Volume` et `Close`. Quel code trace le nuage de points de `Close` en fonction de `Volume` ?

- ☐ `df.plot(x="Volume", y="Close")`
- ☐ `df.scatter(x="Volume", y="Close")`
- ☐ `df["Volume"].plot.scatter(y="Close")`
- ☐ `df.plot.scatter(x="Volume", y="Close")`

12. Dans `fig, ax = plt.subplots(figsize=(8, 4))`, que désignent `fig` et `ax` ?

- ☐ `fig` est l'axe des abscisses, `ax` celui des ordonnées
- ☐ `fig` est la figure entière (l'image), `ax` un repère dessiné dessus
- ☐ `ax` est la figure entière, `fig` l'un de ses repères
- ☐ Les deux sont interchangeables : `fig.plot(...)` et `ax.plot(...)` tracent la même courbe

13. Qu'affiche ce code ?

```
idx = pd.date_range("2020-01-01",
                    periods=100, freq="D")
s = pd.Series(range(100), index=idx)
print(len(s.loc["2020-01-10":"2020-01-20"]),
      len(s.loc["2020-02"]))
```

- ☐ 11 29
- ☐ 10 28
- ☐ 11 28
- ☐ 10 29

14. Le 1er janvier 2021 est un vendredi. Qu'affiche ce code ?

```
idx = pd.date_range("2021-01-01",
                    periods=14, freq="D")
s = pd.Series(1, index=idx)
print(s.resample("W").sum().tolist())
```

- ☐ [7, 7]
- ☐ [14]
- ☐ [2, 7, 5]
- ☐ [3, 7, 4]

15. Que vaut `pd.Series([1, 2, 3, 4, 5]).rolling(3).mean().tolist()` ?

- ☐ [2.0, 3.0, 4.0]
- ☐ [nan, nan, 2.0, 3.0, 4.0]
- ☐ [1.0, 1.5, 2.0, 3.0, 4.0]
- ☐ [nan, nan, nan, 3.0, 4.0]

16. Qu'affiche ce code ?

```
t = pd.Timestamp("2026-10-20")
d = t - pd.Timestamp("2026-09-08")
print(type(d).__name__, d.days)
```

- ☐ Timestamp 42
- ☐ int 42
- ☐ Timedelta 42
- ☐ Timedelta 43

11 ☐ 17. Qu'affiche ce code ?

```
s = pd.Series(pd.to_datetime(["2026-10-20"]))
print(s.dt.day_name()[0], s.dt.month[0])
```

- ☐ Tuesday 10
- ☐ Monday 10
- ☐ Tuesday 20
- ☐ mardi 10

12 ☐

18. Qu'affiche ce code ?

```
df = pd.DataFrame({
    "pays": ["FR", "FR", "IT", "IT"],
    "date": ["d1", "d2", "d1", "d2"],
    "cas": [1, 2, 3, 4]})
t = df.set_index(["pays", "date"])["cas"]
t = t.unstack()
print(t.shape, list(t.columns))
```

- ☐ (4, 1) ['cas']
- ☐ (2, 2) ['FR', 'IT']
- ☐ (4, 3) ['pays', 'date', 'cas']
- ☐ (2, 2) ['d1', 'd2']

13 ☐

19. Soit `y = [1, 2, np.nan, 4, 5]`. Que trace `plt.plot(range(5), y)` ?

- ☐ Une courbe continue, la valeur manquante étant interpolée
- ☐ Deux segments séparés, la ligne étant interrompue au NaN
- ☐ Une erreur `ValueError`
- ☐ Rien : la figure reste vide

17 ☐

18 ☐

19 ☐

20. Qu'affiche ce code ?

```
df = pd.DataFrame({
    "pays": ["FR", "IT", "FR", "IT"],
    "jour": [1, 1, 2, 3],
    "cas": [10, 20, 30, 40]})
s = df.set_index(["pays", "jour"])["cas"]
print(len(s.loc["FR"]), s.loc["FR"].sum())
```

- ☐ 4 100
- ☐ 1 10
- ☐ 2 40
- ☐ Une erreur `KeyError` : il faut donner le couple complet ("FR", 1)

15 ☐

16 ☐

20 ☐