Examen de Niveau

Master 222 : Introduction à Python pour la Finance

Septembre 2024

Instructions pour l'examen

- **Contexte** : Cet examen, d'une durée de 2 heures, vise à évaluer votre niveau en Python. Les étudiants obtenant une note supérieure à 10 pourront, s'ils le souhaitent, être dispensés des trois premiers cours couverts par le programme de cet examen.
- **Aucun appareil** : L'utilisation de calculatrice, d'ordinateur, smartphone ou de tout appareil connecté à Internet est interdite.
- **Documents interdits** : L'utilisation ou la consultation de notes ou tout autre document écrit est strictement interdite pendant l'examen.
- **Langage de programmation** : Tout le code doit être écrit en Python.
- **Notation**: Il y a au total 9 questions notées sur 20 points.

Veuillez lire attentivement les questions et faites de votre mieux.

1 Bases et Fondamentaux (8 points)

Question 1. [3 points] Écrivez une fonction intitulée `count_vowels_and_consonants(s)`, où `s` représente une chaîne de caractères. Cette fonction doit **retourner un dictionnaire** contenant le nombre de voyelles et de consonnes présentes dans la chaîne fournie. Les clés du dictionnaire doivent être `'vowels'` pour les voyelles et `'consonants'` pour les consonnes.

Contraintes supplémentaires :

- La chaîne d'entrée `s` est entièrement en minuscules.

- La chaîne d'entrée ne contient que des caractères alphabétiques (a-z).
- Ignorez les caractères non alphabétiques.

Exemple :

```
>>> count_vowels_and_consonants("bonjour")
{'vowels': 3, 'consonants': 4}
```

Question 2. [1 point] Écrivez une fonction nommée `generate_cubed_numbers(n)` où `n` est un entier. Cette fonction doit renvoyer une liste contenant les cubes des nombres entre `1` et `n` **inclus**.

Indication: Utilisez une compréhension de liste pour générer la liste des cubes.

Exemple :

```
>>> generate_cubed_numbers(5)
[1, 8, 27, 64, 125]
```

Question 3. [1 point] Écrivez une fonction nommée `is_prime(n)` qui accepte un entier `n` comme argument. La fonction doit renvoyer `True` si le nombre est premier et `False` s'il ne l'est pas.

Indication : Un nombre premier est un nombre entier supérieur à 1 qui n'a aucun diviseur positif autre que 1 et lui-même. Utilisez une boucle pour vérifier les diviseurs jusqu'à la racine carrée de ***n****.*

Exemple :

```
>>> is_prime(11)
True
>>> is_prime(12)
False
```

Question 4. [3 points] Écrivez une fonction nommée `fibonacci_memoization(n)` où `n` est un entier. Cette fonction doit renvoyer le $n^{\text{ième}}$ nombre de la séquence de Fibonacci en utilisant un dictionnaire pour optimiser les calculs. La séquence de Fibonacci est définie comme suit :

$$F(0) = 0, \quad F(1) = 1$$

$$F(n) = F(n-1) + F(n-2) \quad \text{pour} \quad n > 1$$

Contraintes supplémentaires :

- Utilisez un dictionnaire pour mémoriser les valeurs déjà calculées de la séquence.
- La fonction doit être capable de gérer des valeurs de `n` élevées (par exemple, $n = 1000$).

2 Numpy et Analyse de Données (6 points)

Question 5. [2 points] Ecrivez une fonction nommée `process_matrix(X)` qui accepte une matrice $n \times n$ comme argument et effectue les opérations suivantes :

1. Calcule et renvoie la somme des éléments de la diagonale principale.
2. Calcule et renvoie la somme des éléments de la sous diagonale secondaire.

Exemple :

```
X = np.array([
    [1, 2, 3, 4],
    [5, 6, 7, 8],
    [9, 10, 11, 12],
    [13, 14, 15, 16]
])
sum_main, sum_secondary = process_matrix(X)
print("Somme de la diagonale principale:", sum_main) # 34
print("Somme de la diagonale secondaire:", sum_secondary)
# 30

```

Question 6. [2 points] Écrivez une fonction nommée `random_symmetric_matrix(n)` où `n` est un paramètre entier. La fonction doit générer une matrice carrée symétrique de dimension $n \times n$. Les éléments hors-diagonaux doivent être égaux à 1, tandis que les éléments sur la diagonale doivent être des nombres générés aléatoirement et uniformément répartis entre 0 et 1.

Question 7. [2 points] Créez un tableau numpy `W` contenant 30 valeurs aléatoires comprises entre 50 et 150. Déterminez et affichez la médiane et l'écart-type des éléments de ce tableau.

Indication : Utilisez la fonction ***`np.median()`*** *de numpy pour calculer la médiane et* ***`np.std()`*** *pour obtenir l'écart-type.*

3 Pandas et Matplotlib (6 points)

Question 8. [4 points] Créez un DataFrame `sales_data` via pandas à partir des informations du marché fournies ci-dessous, puis effectuez les opérations suivantes :

```
    data = {
        'Instrument': ['A', 'B', 'C', 'D'],
        'Sales Volume': [100, 150, 80, 120],
        'Price per Unit': [10.5, 20.0, 15.75, 30.0]
    }

```

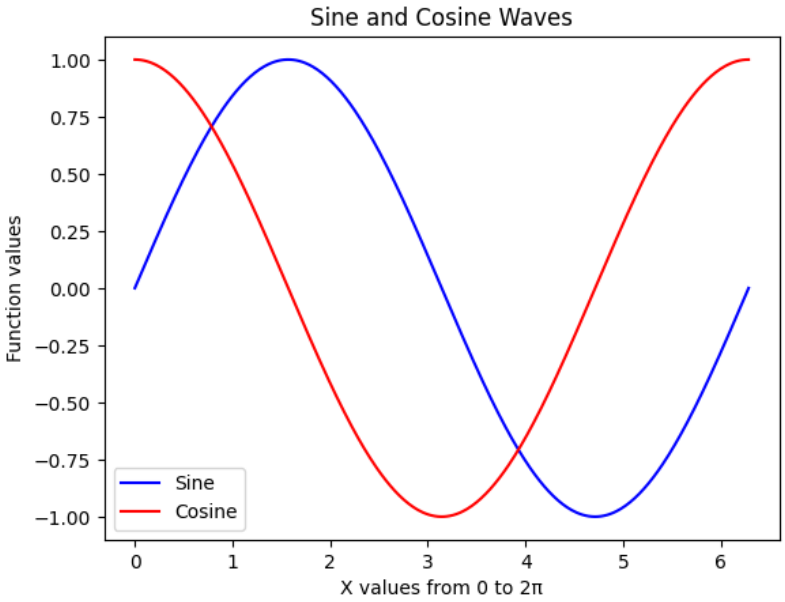
1. Calculez et ajoutez une nouvelle colonne `'Total Sales'` au DataFrame qui représente le produit du `'Sales Volume'` et du `'Price per Unit'` pour chaque instrument.

2. Calculez et présentez le volume total de transactions pour l'ensemble des instruments financiers.

3. Calculez et présentez la valeur totale des ventes pour l'ensemble des instruments financiers.

Question 9. [2 points] Rédigez le code permettant de représenter le graphique ci-après :

La courbe de la fonction 'Sinus' est colorée en bleu, tandis que celle de la fonction 'Cosinus' est en rouge.


Sine and Cosine Waves
Function values
X values from 0 to 2π
Sine
Cosine
1.00
0.75
0.50
0.25
0.00
−0.25
−0.50
−0.75
−1.00
0
1
2
3
4
5
6