Examen de Niveau

Master 222 : Introduction à Python pour la Finance

Septembre 2025

Instructions pour l'examen

— **Contexte** : Cet examen, d'une durée de 1h30, 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.

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.

Hypothèse pour la chaine d'entrée :

— La chaîne d'entrée `s` est entièrement en minuscules.
— La chaîne d'entrée ne contient que des caractères alphabétiques (a-z).

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.

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` sinon.

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(n)` où `n` est un entier. Cette fonction doit renvoyer le n-ième nombre de la séquence de Fibonacci en utilisant un dictionnaire pour optimiser les calculs. La séquence est définie par :

$$F(0) = 0, \quad F(1) = 1, \quad F(n) = F(n-1) + F(n-2), \quad \text{pour } n > 1$$

Contraintes :

— Utilisez un dictionnaire pour mémoriser les valeurs déjà calculées.
— La fonction doit pouvoir gérer des valeurs élevées (ex. `n = 1000`).

2. Numpy et Analyse de Données (6 points)

Question 5. **2 points**

Écrivez une fonction nommée `process_matrix(X)` qui accepte une matrice `n × n` et effectue les opérations suivantes :

1. Calcule et retourne la somme des éléments de la diagonale principale.
2. Calcule et retourne 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)
# Somme de la diagonale principale: 34
# Somme de la diagonale secondaire: 30
```

Question 6. **2 points**

Écrivez une fonction nommée `random_symmetric_matrix(n)` où `n` est un entier. La fonction génère une matrice carrée symétrique `n × n` avec :

— des 1 en dehors de la diagonale principale
— des valeurs aléatoires uniformes entre 0 et 1 sur la diagonale principale

Indications : Utilisez `np.random.uniform()` et `np.array()`

Question 7. **2 points**

Créez un array `numpy` `W` contenant 30 valeurs aléatoires suivant une loi uniforme -1 et 1. Déterminez et affichez la médiane et l'écart-type avec la fonction `print()`.

Indications : Utilisez `np.random.uniform()`, `np.median()`, `np.std()`

3. Pandas et Matplotlib (6 points)

Question 8. **4 points**

Créez un `DataFrame` `df` via `pandas` à partir du dictionnaire suivant :

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

Puis effectuez les opérations suivantes :

1. Calculez et ajoutez une colonne `'Total Sales'` (produit du volume et prix unitaire).
2. Calculez la somme du volume total de transactions puis l'afficher avec la fonction `print()`.
3. Calculez la somme de la valeur totale des ventes puis l'afficher avec la fonction `print()`.

Question 9. **2 points**

Écrivez un code python permettant de représenter avec `matplotlib` :

— La courbe de la fonction `Sinus` en bleu entre $-\pi$ et π
— La courbe de la fonction `Cosinus` en rouge entre $-\pi$ et π

Indication : Utilisez `x = np.linspace(-np.pi, np.pi, 500)`, `np.sin(x)`, `np.cos(x)`, `plt.plot()`