EXERCICES

Python — Programmation orientée objet et récursivité

1re

Nom : _____
Classe : _____

> Au contrôle, une réponse juste sans calcul ne rapporte presque rien : chaque problème se rédige en quatre temps — **justifier**, **poser** l'égalité, **mener le calcul** jusqu'au bout, **conclure** par une phrase avec l'unité. La place réservée est dimensionnée pour ces quatre temps.

1 Vocabulaire. Répondre en une phrase précise. ●○○

a. Que fait exactement une affectation de variable ?
b. Différencier une classe et une instance.
c. Qu'est-ce que l'encapsulation ? Quel est son intérêt réel ?
d. Quels sont les deux éléments indispensables d'une fonction récursive ?
e. À quoi sert une exception ?

2 Une fonction reçoit une liste et y ajoute un élément. L'appelant constate que sa propre liste a changé. ●●○

a. Pourquoi la liste de l'appelant a-t-elle changé ?
b. Le même phénomène se produirait-il avec un entier ? Pourquoi ?
c. Comment obtenir le comportement inverse, si on ne le voulait pas ?
d. Pourquoi ce comportement surprend-il souvent les débutants ?

3 Récursivité. Raisonner sur la terminaison et le coût. ●●○

a. Écrire en français le cas de base et le cas général de la factorielle.
b. Calculer la factorielle de 4 en déroulant.
c. Que se passe-t-il si le cas de base est oublié ?
d. Une récursivité naïve sur Fibonacci est-elle correcte ? Est-elle utilisable ?
e. Que faut-il retenir de la distinction entre ces deux propriétés ?

4 Objet et exceptions. Analyser.

●●●

a. Un carré doit-il hériter de rectangle ? Un moteur et une voiture ?

b. Pourquoi l'encapsulation permet-elle de garantir un état cohérent ?

c. Pourquoi ne faut-il pas rattraper toutes les exceptions sans rien en faire ?

d. Pourquoi importer un module plutôt que réécrire une fonction ?

e. Ce dernier argument vaut-il sans réserve ?

Corrigé détaillé sur **progresschool.fr** — chapitre « Python — Programmation orientée objet et récursivité ».