NSI · TERMINALE / EXERCICES CORRIGÉS

Les paradigmes de programmation

Astrolab Cours · astrolabcours.fr

Ex. 1 Pure ou pas ●○○ Application

Ces fonctions sont-elles pures ?

```
def f(x):
    return 2 * x + 1

def g(liste):
    liste.append(0)
    return len(liste)
```

Indice

Modifie-t-elle quelque chose à l'extérieur ?

f

f ne dépend que de x et ne modifie rien : elle est pure.

g

g n'est pas pure : elle modifie la liste reçue (effet de bord), et deux appels sur la même liste donnent des longueurs différentes.

Ex. 2 Réécrire en fonctionnel ●●○ Méthode

Réécrire avec `map` et `filter` :

```
res = []
for mot in mots:
    if len(mot) > 3:
        res.append(mot.upper())
```

Indice

D'abord filtrer, puis transformer.

© Astrolab Cours · astrolabcours.fr/nsi/terminale/paradigmes-de-programmation/ · Tous droits réservés : usage personnel uniquement, republication et usage commercial interdits

La version fonctionnelle

```
longs = filter(lambda m: len(m) > 3, mots)
res = list(map(lambda m: m.upper(), longs))
```

Ex. 3 Le map invisible ●●○ Piège

Un élève écrit `print(map(lambda x: x + 1, [1, 2, 3]))` et obtient `<map object at 0x...>`. Pourquoi ?

Indice

Que renvoie `map` ?

L'erreur

`map` ne renvoie pas une liste mais un objet qui produit les valeurs à la demande.

La correction

`print(list(map(lambda x: x + 1, [1, 2, 3])))` affiche [2, 3, 4].

Ex. 4 Une fonction en paramètre ●●● Type bac

Écrire une fonction `appliquer_deux_fois(f, x)` qui renvoie f(f(x)). Que vaut `appliquer_deux_fois(lambda n: n * 3, 2)` ?

Indice

Une fonction est une valeur comme une autre.

La fonction

```
def appliquer_deux_fois(f, x):
    return f(f(x))
```

Le calcul

f(2) = 6, puis f(6) = 18.

© Astrolab Cours · astrolabcours.fr/nsi/terminale/paradigmes-de-programmation/ · Tous droits réservés : usage personnel uniquement, republication et usage commercial interdits