

Corrigé du QCM n°2

1. Soit f qui réalise $x \mapsto x^2$:

```
def f(x):  
    res=x**2  
    ...
```

Les points ... sont à remplacer par

1. `return [res]`
2. `res`
3. `print(res)`
4. `return res`

La fonction doit renvoyer le résultat stocké dans `res` d'où `return res`.

2. Après l'instruction `a=list(range(1,5))`, on a :

1. la variable `a` contient `[0,1,...,4]`
2. la variable `a` contient `[1,...,4]`
3. la variable `a` contient `[1,...,5]`
4. rien car l'instruction provoque une erreur

On construit la séquence de 1 à 5 exclu par pas de 1 et on la convertit en liste.

3. Après l'instruction `a=list(range(1,11,2))`, on a :

1. la variable `a` contient `[1,2,...,9]`
2. la variable `a` contient `[1,3,...,9]`
3. rien car l'instruction provoque une erreur
4. la variable `a` contient `[1,2,...,10]`

On construit la séquence de 1 à 11 exclu par pas de 2 et on la convertit en liste.

4. Un argument d'une fonction :

1. n'est ni une variable globale, ni une variable locale
2. est éphémère, ne vivant que dans la fonction
3. est une variable globale
4. est une variable locale

Un argument est un cas particulier de variable locale donc éphémère, ne vivant que dans la fonction considérée.

5. Une variable locale d'une fonction :

1. peut être utilisée pour modifier une variable globale
2. peut être en conflit avec une variable globale
3. peut être utilisée hors de la fonction
4. peut avoir le même nom qu'une variable globale

Une variable locale n'existe que dans le corps d'une fonction et n'est jamais en conflit avec une variable globale car en cas d'homonymie, la fonction ne voit que la variable locale. Si l'argument de la fonction est une liste, cet argument est une variable locale étiquetée sur une liste qui est mutable et peut donc être modifiée depuis la fonction.

6. Pour tester une fonction saisie dans un script dans l'éditeur :

1. on peut se passer totalement de la console
2. on peut programmer des affichages de résultats de la fonction dans le script
3. on peut l'utiliser dans la console après avoir exécuté le script
4. on peut l'utiliser dans la console sans exécuter le script

Pour tester une fonction saisie dans un script, on peut exécuter le script dans la console pour ensuite appeler cette fonction ou alors programmer directement dans le script des affichages de résultats de la fonction qui seront affichés dans la console.

7. Si l'on modifie une fonction saisie dans un script dans l'éditeur :

1. la modification est immédiatement effective dans la console
2. il faut éteindre l'ordinateur puis le rallumer et relancer python pour rendre une modification effective
3. il faut fermer la console après chaque modification pour les rendre effectives
4. la modification n'est effective qu'après avoir ré-exécuter le code

La modification d'une fonction n'est effective qu'après avoir ré-exécuter le code (touche F5).

8. Une fonction :

1. peut contenir une fonction
2. renvoie systématiquement un résultat
3. peut mettre le feu à l'ordinateur et il faut donc faire preuve de prudence avant de la tester
4. peut avoir une fonction comme argument

Une fonction peut prendre une autre fonction comme argument et peut contenir une fonction locale.

9. L'instruction `if condition:`

1. n'existe pas
2. est suivie d'une indentation
3. sert à coder un test
4. sert à coder une boucle

L'instruction `if condition:` sert à coder un test et les instructions réalisées sous conditions sont indentées.

10. L'instruction `for . . . . : :`

1. peut être utilisée pour parcourir un `range`
2. sert à coder un test
3. n'existe pas
4. sert à coder une boucle

L'instruction `for . . . . : :` sert à coder une boucle et peut être utilisée pour parcourir un `range`.