ISM MPSI/PCSI, Informatique
Année 2024/2025

Corrigé du TP Informatique 1

Exercice 1

1. Python applique ses règles de précédence. Dans le doute, on recommande le bon usage des parenthèses.
2. Les entiers, aussi grands soient-ils en valeur absolue, sont du type `int`.
3. Avec les opérations de reste et quotient, on peut décomposer l'écriture décimale d'un nombre.

Exercice 2

1. L'opération `/` bascule dans les flottants, même pour un résultat entier. De plus, le format `float` est absorbant.
3. La lettre `e` désigne une puissance de 10 et est en format flottant. L'exponentiation d'un entier par un entier naturel est en format entier. 4. L'arithmétique flottante n'est pas de l'arithmétique formelle : avoir une confiance mesurée dans l'exactitude du calcul numérique.

Exercice 3

1. La conversion en entier tronque la partie décimale.
2. La conversion en flottant est contrainte par les limitations du format.

Exercice 4

1. Les objets à deux états `True`, `False` sont des booléens, de type `bool`.
2. Python applique ses règles de précédence. Dans le doute, on recommande le bon usage des parenthèses.
3. Le résultat d'un test est un booléen.

Exercice 5

1,2. Python confond `True` avec 1 et `False` avec 0. Une quantité nulle est convertie en `False` et non nulle en `True`.

Exercice 6

Sans surprise, les deux premières instructions renvoient une erreur. Mais la dernière renvoie `True`. Ceci est la conséquence de l'évaluation paresseuse : le début de l'expression détermine le résultat donc python ne prend pas la peine d'évaluer la fin de l'expression (c'est une fonctionnalité bien commode en programmation ...).

Exercice 7

1. Le calcul flottant n'est pas du calcul formel : des nombres proches de 10^{-16} peuvent être interprétés comme nuls.
2. Les fonctions $\sqrt{\cdot}$ et $\lfloor\cdot\rfloor$ sont à valeurs dans les flottants.

3. La partie entière et la conversion en entière renvoient des valeurs décalées pour les nombres négatifs.

Exercice 8

1. Le typage en python est dynamique : une variable peut changer de type si on lui affecte un objet d'un nouveau type.
2. On peut modifier directement le contenu d'une variable avec des instructions de type `+=`, `//=`, etc.

Exercice 9

1. Avec `a,b= ...`, on peut faire des affectations simultanées. Le couple `a,b` est une *target list*.
2. De même, on peut utiliser une target list pour échanger des variables, sans usage d'une variable auxiliaire. C'est très confortable.

Exercice 10

L'instruction `print(..)` permet l'affichage d'un ou plusieurs objets à la suite.

Exercice 11

On saisit :

```
a=153
print(a==(a%10)**3+((a//10)%10)**3+(a//100)**3)
```

Exercice 12

On saisit :

```
a=1551
print((a%10)==(a//1000) and (a//10)%10==(a//100)%10)
```