Les types

**Enoncé :**

Le professeur Rogue a demandé à Rohn de travailler à la confection d’une potion donnant le pouvoir de faire taire quiconque aurait le malheur de la boire. Rohn a besoin de ton aide : il a horreur des conversions, et tout est indiqué en louche alors qu’il n’a à sa disposition qu’une petite cuillère !

Une louche contient l'équivalent de 6 petites cuillères.

La recette indique :

*Pour 1 fiole de potion silencieuse :*

* *2 louches de sève d’harmonie*
* *½ louche d’élixir de quiétude*
* *¼ louche de nectar de tranquillité*

Rohn doit convertir ces quantités en cuillères, une étape cruciale pour réussir la potion. Heureusement tu proposes de créer un sort « le silencio » pour l’aider à réaliser sa potion.

1. **Niveau 0 :**

Écris un programme qui renvoie la quantité nécessaire de nectar de tranquillité en nombre de petites cuillères (ce nombre peut être décimal).

1. **Niveau 1 :**

Modifie ton programme qui renvoie la quantité nécessaire de nectar de tranquillité en nombre de petites cuillères, pour qu’elle soit cette fois arrondi au chiffre supérieur (il vaut toujours mieux en avoir trop que pas assez). Ce nombre doit être un entier.

Remarque :

Pour effectuer des opérations mathématiques de base en python, tu peux importer un module à l’aide de « from *nom\_du\_module* import *nom\_de\_la\_fonction* ». Par exemple, le module « math » contient les fonctions « ceil » qui permet d’arrondir au chiffre supérieur, « floor » qui permet d’arrondir au chiffre supérieur, « sum », etc.

Pour importer l’ensemble des fonctions contenue dans un module tu peux utiliser le symbole « \* » à la place du nom de la fonction.

1. **Niveau 2 :**

Écris un programme qui renvoie l’ensemble de la recette convertie en utilisant la petite cuillère comme unité de mesure. Le programme doit retourner une chaîne de caractères de la forme :

----------

Pour 1 fiole de potion silencieuse :

* 12 petites cuillères de sève d’harmonie
* 3 petites cuillères d’élixir de quiétude
* 2 petites cuillères de nectar de tranquillité

----------

1. **Niveau 3**

Modifie ton programme pour qu’il soit facilement possible de modifier la quantité finale de potion souhaitée en nombre de fioles, puis affiche ensuite la recette avec les quantités originales en louche et leurs conversions en petites cuillères, sous forme d’une chaîne de caractères.

Par exemple :

----------

Combien de fioles de potion silencieuse souhaitez-vous créer ? 3

Pour 3 fioles de potion silencieuse :

- 8 louches soit 36 petites cuillères de sève d’harmonie

- 9 louches soit 22 petites cuillères d’élixir de quiétude

- 6 louches soit 12 petites cuillères de nectar de tranquillité

----------

1. **Niveau 4**

Modifie ton programme pour qu’il soit également possible de modifier la capacité d’une louche en petite cuillères.

Que se passerait-il si Rohn avait oublié une partie de la recette ? Peux-tu facilement modifier ton programme pour qu’il soit capable de gérer un tel cas ?

Par exemple si la recette indiquait plutôt :

----------

Pour 1 fiole de potion silencieuse :

* 2 louches de sève d’harmonie
* ½ louche d’élixir de quiétude
* ¼ louche de nectar de tranquillité
* 1 louches de sirop de framboise

----------

Tips : Tu peux regarder comment utiliser des listes et faire des itérations sur une liste en python.

**Pour aller plus loin :**

1. **Typer ses variables**

Python est un langage typé dynamique. C’est-à-dire que lorsque tu utilises une nouvelle variable, il n’est pas nécessaire de la déclarer ni de spécifier son type, Python « devine » si la nouvelle variable est un entier ou une chaine de caractère par exemple. Cela n'est pas le cas pour de nombreux autres langages (comme le C et le C++, par exemple). Apprendre à bien déclarer et à typiser ses variables est une très bonne pratique qui permet notamment de garantir la cohérence et la robustesse du code.

Pour cela, lorsque tu souhaites utiliser une nouvelle variable comme "a", par exemple, plutôt que d'écrire simplement : a = 6 + 3

Écris plutôt :

a: int

a = 6 + 3

Ceci spécifie explicitement que la variable "a" est de type entier (int). Cette pratique de typage explicite rend le code plus lisible et permet aux autres développeurs (ou à toi-même dans le futur) de mieux comprendre le code sans avoir à deviner les types des variables.

1. **Mypy : un vérificateur de type**

Si tu es curieux·se, tu peux utiliser **mypy**, qui est un vérificateur de type pour Python. Il te dira à quel endroit tu n’as pas correctement déclaré tes variables.

Pour installer **mypy** :

Une image contenant capture d’écran, Police, symbole, blanc

Description générée automatiquementOuvre un terminal sur ton ordinateur. Tu peux aussi y accéder directement depuis ton IDE (comme PyCharm ou Visual Studio Code). Recherche une icône qui ressemble à cela :

Écris "pip install mypy". Si tu rencontres une erreur indiquant que pip n’a pas été trouvé, installe pip en suivant un tutoriel en ligne. Cela te permettra par la suite d'installer facilement d'autres packages.

Comment utiliser **mypy** :

Rends-toi dans le bon répertoire avec la commande **cd** suivi du chemin d’accès au fichier. Une fois au bon endroit écris "mypy nom\_de\_ton\_fichier.py" dans le terminal pour vérifier le fichier Python spécifié.