Mini-projet
Harry se fait la malle !

I. Sur le chemin de Poudlard

1. La monnaie des sorciers

Harry a reçu sa lettre d'admission pour l'école de Poudlard. Dans l'enveloppe il était ajouté la liste de fournitures scolaires à apporter pour la rentrée.

Hagrid vient le chercher et l'accompagne chez Gringotts, la banque des sorciers, pour retirer de l'argent.

Dans le monde magique, le système de monnaie est le suivant :

- 1 Gallion (pièce en or) vaut 17 Mornilles (pièce en argent)
- 1 Mornille vaut 29 Noises (pièce en bronze)

2. Le chemin de traverse

Avec une bourse bien garnie, Harry est prêt pour découvrir le chemin de traverse, lieu commerçant propre aux achats divers. Harry va pourvoir faire ses achats pour la rentrée chez plusieurs artisans.

a. L'achat de la robe de sorciers

Après avoir essayé et choisi une robe de sorcier réglementaire, Harry paye mais il dispose de peu de monnaie et il n'est pas encore très habitué à ce système. Dans le doute, il donne donc sa plus grosse pièce : un Gallion.

Le patron de l'échoppe est en fait un sorcier qui travaille secrètement pour les moldus (humains non sorciers). Il dispose donc de monnaie moldu qu'il a parfois du mal à utiliser dans le monde magique : les euros.

Même s'il est parfaitement honnête, en rendant la somme due sur le Gallion donné par Harry, ce sorcier en profite pour écouler sa monnaie moldu.

- Élaborer un algorithme permettant de déterminer toutes les pièces et billets à rendre à Harry en euros (sans tenir compte des centimes), d'après la somme totale à rendre.
- Écrire cet algorithme en pseudo-code.
- Programmer cet algorithme en Python.

Remarques :

- Peu importe le prix de la robe, on ne donne et on ne s'intéresse qu'à la somme à rendre par le patron de l'échoppe.
- L'algorithme doit s'adapter à n'importe quelle somme à rendre.
- On considérera le tiroir-caisse de l'artisan infini, c'est à dire qu'aucune pièce ou billet ne peut venir à manquer.
- L'utilisation de types construits est obligatoire.
- L'utilisation de fonction est obligatoire

Tester votre algorithme, puis votre programme, avec les sommes à rendre suivantes : 60 €, 62 €, 63 €, 231 € et 239 €.

Très satisfait de votre visite, le patron vous invite à passer chez son voisin pour acheter un chaudron.

b. L'achat du chaudron

Après les présentations d'usage, le vendeur de chaudron oriente directement Harry vers un magnifique chaudron en cuivre. Quel chance, il est en promotion !

Comme son voisin, le patron propose de lui rendre la monnaie en euros mais, cette fois-ci, le tiroir caisse est limité :

- 1 billet de 200 €
- 3 billets de 100 €
- 1 billet de 50 €
- 2 billets de 20 €
- 1 billet de 10 €
- 2 billets de 5 €
- 5 pièces de 2 €

Tester votre algorithme, puis votre programme, avec les sommes à rendre suivantes : 60 €, 62 €, 63 €, 231 € et 239 €.

Élaborer un nouvel algorithme et un nouveau programme pour rendre la monnaie à Harry.

Il est probable que votre algorithme ne permette pas un rendu optimal (somme exacte à rendre). Dans ce cas, le vendeur devra rendre davantage d'argent à Harry (le minimum possible tout de même) et votre algorithme devra le signifier.

c. L'achat de la baguette magique

La baguette qu'a choisi Harry (ou plutôt la baguette « qui a choisi » Harry !) vaut 2 Gallions, 5 Mornilles et 20 Noises.

Dans l'excitation et la précipitation Harry donne 5 Gallions au fabriquant de baguettes.

Élaborer un nouvel algorithme et un nouveau programme pour satisfaire ce nouveau système de monnaie.

Tester votre algorithme, puis votre programme, sur les sommes à rendre suivantes :

- 34 Noises
- 654 Noises
- 2687 Noises
- 23 Mornilles et 78 Noises
- 2 Gallions, 11 Mornilles et 9 Noises

3. Cahier des charges

Votre programme doit être conçu pour faciliter son test et son évaluation.

Il doit donc être complet, exécutable immédiatement avec un interpréteur Python.

Vos fonctions doivent être parfaitement documentées dans le docstring.

Votre IHM doit nécessairement :

- automatiser les tests de rendu de monnaie imposés ci-dessus.
  - L'utilisation de doctest dans le docstring peut-être une solution élégante pour ces tests (voir notebook 2_10A sur les fonctions avancées), mais cette solution n'est pas exigible.
  - Ces tests peuvent être exécutés au lancement du programme ou à l'initiative de l'utilisateur, via une IHM agréable.
- permettre de faire des achats dans les trois échoppes.
  - L'utilisateur doit pouvoir saisir une somme à rendre et l'IHM affichera les détail de la monnaie rendue.

L'IHM ne doit pas nécessairement passer par une interface graphique de type Tkinter, même si cette option est parfaitement envisageable.

Les algorithmes doivent être rédigés en pseudo-code. Ils peuvent être intégrés au programme, via l'IHM, ou être rendus dans un document séparé, au format PDF.

Enregistrer le programme sous le format : Malle_1_NOM1_NOM2_NOM3.py

Si vous avez un document PDF en plus… Malle_1_NOM1_NOM2_NOM3.pdf

Inutile de créer un fichier compressé si vous n'avez que un ou deux fichiers à rendre.

Votre projet doit être déposé sur l'espace de travail NSI d'e-lyco (« Vue d'ensemble », « tâches »), en temps et en heure. Vous devrez y créer votre groupe avant de déposer vos fichiers.

4. Évaluation

Critères d'évaluation (note sur 10 points) :

- Respect du cahier des charges (5 points)
- Bonnes pratiques PEP-8 (dont les docstrings) (2 points)
- Travail d'équipe (1 point)
- Utilisation d'outils collaboratifs numériques (1 point)
- Point « subjectif » (1 point)
- Bonus (+ 2 points envisageables en fonction de l'ampleur du bonus)

II. Derniers préparatifs avant le départ pour Poudlard

Avant le départ, Harry doit préparer sa malle.

A l'issue de la première étape (Chemin de traverse), nous verrons comment l'aider.