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. Chez Fleury & Bott, libraire

Après avoir trouvé les livres nécessaires pour sa rentrée (Le Livre des sorts et enchantements, niveau 1 de Miranda Fauconnette, Histoire de la magie de Bathilda Tourdesac), 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 libraire 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 des livres, on ne s'intéresse qu'à la somme à rendre par le libraire.
- L'algorithme doit rendre la monnaie avec le moins de billets et de pièces possible.
- Les pièces et billets disponibles ont pour valeur : 1, 2, 5, 10, 20, 50, 100, 200 et 500 €.
- L'algorithme doit s'adapter à n'importe quelle somme à rendre.
- On considérera le tiroir-caisse de capacité infinie, 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 : 0€, 60 €, 63 €, 231 € et 899 €.

Très satisfait de votre visite, le patron vous invite à passer chez son voisin pour acheter une robe de sorcier.

b. Chez Madame Guipure, prêt-à-porter pour mages et sorciers

Après les présentations d'usage, Mme Guipure oriente Harry vers une magnifique robe de travail noire. Quel chance, elle est en promotion !

Comme son voisin, Mme Guipure 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 €
- 1 billets de 20 €
- 1 billet de 10 €
- 1 billet de 5 €
- 5 pièces de 2 €

Tester votre algorithme, puis votre programme, avec les sommes à rendre suivantes : 0€, 17 €, 68 €, 196 €, 595 € et 842 €.

Votre algorithme est-il en en mesure de gérer les rendus de monnaie sur de grosses sommes ?

Votre algorithme permet-il un rendu parfait dans tous les cas (somme exacte à rendre) ?

Élaborer un nouvel algorithme et un nouveau programme pour rendre la monnaie à Harry dans ce cas de tiroir caisse limité. Votre algorithme devra signaler par un message explicite si une situation défavorable se présente (ex : rendu de monnaie incomplet).

Bonus : Dans le cas où votre algorithme ne permet pas de rendre la totalité de la somme, et si il reste de l'argent en caisse, Mme Guipure devra rendre davantage d'argent à Harry (le minimum possible tout de même). Votre algorithme devra signaler cette situation par un message clair.

c. Chez Ollivander, fabriquant de baguettes magiques

Dans la boutique de Garick Ollivander, les baguettes magiques sont nombreuses et variées.

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 à M. Ollivander.

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

A nouveau, **nous ne nous intéressons qu'à la somme à rendre, en un minimum de pièces**.

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

- 0 Noises
- 654 Noises
- 23 Mornilles et 78 Noises
- 2 Gallions, 11 Mornilles et 9 Noises
- 7 Gallions, 531 Mornilles et 451 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.

Il est conseillé de construire une fonction par objectif (libraire, robe et baguette magique). Il faut privilégier les « vraies » fonctions au procédures (une fonction renvoie quelque chose). 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), mais cette solution n'est pas exigible.
  - Ces tests peuvent être exécutés au lancement du programme ou bien à l'initiative de l'utilisateur, via une IHM agréable.
- permettre de faire des achats dans les trois boutiques.
  - 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 ou Pygame, même si cette option est parfaitement envisageable (bonus).

Les algorithmes doivent être rédigés en pseudo-code et ne doivent décrire que le cœur de la problématique (le rendu de monnaie), pas l'IHM. 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 qu'un ou deux fichiers à rendre. Dans le cas où vous auriez crée un dossier, compresser l'ensemble de vos fichiers sous la forme Malle_1_NOM1_NOM2_NOM3.zip pour conserver l'arborescence.

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 lors du dépôt de fichiers.

4. Bonus

Élaborer un algorithme permettant de rendre la monnaie de façon optimale, en toute situation. Il faut, là encore, minimiser le nombre de pièces et billets à rendre. Pour ce bonus, vous pouvez vous baser sur le tiroir-caisse limité de Mme Guipure.

5. Évaluation

Critères d'évaluation (note sur 10 points) :

- Respect du cahier des charges (dont l'IHM) (6 points)
- Algorithmes en pseudo-code (1,5 point)
- Bonnes pratiques PEP-8 (dont commentaires et docstrings) (1 point)
- Implication, organisation du travail en équipe / Utilisation d'outils numériques (1 point)
- ½ Point « subjectif » (0,5 point)
- Bonus (+ 2 points envisageables en fonction de l'ampleur du bonus)

II. Harry se fait (enfin) la malle !

1. Derniers préparatifs avant le départ pour Poudlard

Harry a effectué tous ses achats. Maintenant qu'il est parfaitement équipé, il doit préparer sa malle avant le départ pour l'école de Poudlard.

Il dispose bien d'une malle pour ranger ses affaires mais cette malle est trop fragile pour supporter un poids important. Il ne pourra donc pas mettre toutes ses affaires dans la malle.

Voici la liste de fournitures scolaire des élèves de première année :

- 8 manuels scolaires :
  - Le Livre des sorts et enchantements (niveau 1), de Miranda Fauconnette
  - Histoire de la magie, de Bathilda Tourdesac
  - Magie théorique, de Adalbert Lasornette
  - Manuel de métamorphose à l'usage des débutants, de Emeric G. Changé
  - Milles herbes et champignons magiques, de Phyllida Augirolle
  - Potions magique, de Arsenius Beaulitron
  - Vie et habitat des animaux domestiques
  - Forces obscures : comment s'en protéger de Quentin Jentremble
- 1 baguette magique
- 1 chaudron (modèle standard en étain, taille 2)
- 1 boite de fioles en verre ou cristal, varie également
- 1 télescope
- 1 balance en cuivre
- 3 robes de travail (noires)
- un chapeau pointu (noir)
- une paire de gants protecteurs (en cuir de dragon ou autre matière semblable)
- une cape d'hiver (noire avec attaches d'argent)

Un fichier est mis à votre disposition : les fournitures y sont disponibles dans une table de données. Chaque fourniture possède plusieurs caractéristiques (descripteurs) :

- son nom
- son poids (en kg)
- sa mana, c'est à dire son potentiel magique

Le fichier contient également le poids maximal supporté par la malle.

Remarques :

- Pour simplifier ce projet, on ne considérera qu'un seul exemplaire par objet à emporter.
- Un simple copier / coller de cette liste de fournitures dans votre code est demandé.

2. Comment remplir cette malle ?

a. Remplir la malle… n'importe comment !

Le premier objectif est de remplir la malle le plus vite possible, sans méthode particulière, peu importe ce qu'elle contient, tout en respectant le poids maximal (inclus) qu'elle peut supporter.

Votre programme devra afficher le contenu de la malle et son poids.

b. Remplir la malle la plus lourde possible

Vous devez maintenant concevoir un algorithme puis un programme qui remplira la malle en maximisant le poids des fournitures embarquées, sans toutefois dépasser le poids maximal supporté par la malle. Votre programme devra afficher le contenu de la malle et son poids.

c. Remplir la malle avec le maximum de mana (énergie magique)

Vous devez maintenant concevoir un algorithme puis un programme qui remplira la malle en maximisant l'énergie magique (mana) des fournitures embarquées, sans toutefois dépasser le poids maximal supporté par la malle.

Votre programme devra afficher le contenu de la malle, sa valeur magique totale et son poids.

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.

Si vous avez besoin d'effectuer un tri, vous êtes autorisé à utiliser, au choix, le tri natif (built-in) de Python sort() ou votre propre tri. Inutile de fournir le pseudo-code de ce tri.

Il est à nouveau conseillé de construire une fonction par objectif et une fonction pour le tri. Vos fonctions doivent être parfaitement documentées dans le docstring. Là encore, il est préférable d'utiliser des fonctions plutôt que des procédures, autant que possible.

Votre IHM doit nécessairement automatiser les remplissages de malle, avec une visualisation claire des résultats obtenus.

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

Les algorithmes doivent être rédigés en pseudo-code et ne doivent décrire que le cœur de la problématique (le remplissage de malle), pas l'IHM.

Enregistrer le programme sous le format : Malle_2_NOM1_NOM2_NOM3.py

Si vous avez un document PDF (ex : pseudo-code)… Malle_2_NOM1_NOM2_NOM3.pdf

Inutile de créer un fichier compressé si vous n'avez qu'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 voulu. Vous devrez y créer votre groupe avant de déposer vos fichiers.

4. Bonus possible

- Améliorer le remplissage de la malle en cherchant le meilleur rapport mana / poids des objets contenus dans la malle, afin d'obtenir un bon compromis.
- Ajouter un descripteur « Quantité » aux enregistrements de la table. Tenez compte du fait qu'il y a 8 manuels scolaires disponibles et 3 robes de travail. Adapter l'un ou autre des précédents programme à cette nouvelle contrainte.
- Ajouter une méthode par « force brute » ou, pour des élèves déjà bien aguerris en programmation, par programmation dynamique. Ces deux méthodes apportent une solution optimale dans tous les cas (peu importe les objets à disposition).

5. Évaluation

Idem partie précédente...