Nom:

Prénom :

Aménagement glouton

Point du programme abordés : Algorithmes gloutons

Durée : 3h

/!\ Avant de débuter l'activité :

Créer un dossier Algorithmes_gloutons dans le dossier algorithmique, puis y copier/coller ce fichier et le fichier intitulé glouton.py.

Contexte :

Animal Crossing : New Horizons est un jeu de type bac à sable sorti sur Nintendo Switch en 2020. Dans ce jeu, le joueur possède une maison qu'il peut décorer avec les objets qu'il fabrique ou trouve tout au long de son aventure, maison que l'on peut découper en 6x6 cases (soit 36 cases), semblable à :

Vous êtes un joueur débutant et possédez de ce fait la liste d'objets suivante :

Noms	Dimensions	Nombres de cases	Points d'utilité
Lit double	3x3	9	50
Bureau	2x3	6	20
Bibliothèque	2x3	6	5
Commode	2x3	6	2
Lampe	2x2	4	10
Globe terrestre	2x2	4	1

Noms	Dimensions	Nombres de cases	Points d'utilité
Guéridon	2x2	4	5
Petite table	2x2	4	5
Vase	2x2	4	1
Coffre en bois	2x2	4	2

À chaque objet sont attribuées des dimensions et des points d'utilités.

-> À l'aide des supports physiques à votre disposition, faites des simulations d'aménagement de votre maison.

1. Quelle est la liste d'objets maximisant le nombre de points d'utilités avec une maison de dimensions 6x6 ?

Nombre de cases vides	Points d'utilités maximum	Nombre de cases vide restantes
36		

2. Quelle stratégie adopter pour remplir la pièce d'objets qui MAXIMISENT le nombre de points d'utilités ?

3. Quelle est la liste d'objets maximisant le nombre de points d'utilités avec une maison de dimensions 5x5?

Nombre de cases vides	Points d'utilités maximum	Nombre de cases vide restantes

Nombre de cases vides	Points d'utilités maximum	Nombre de cases vide restantes

Nous voulons implémenter un algorithme qui permet de déterminer cette solution.

4. Quelle structure de données utiliser pour représenter un objet ?

5. Quelle structure de données utiliser pour représenter les différents objets disponibles ?

-> Soit les fonctions suivantes :

```
def points_d_utilites (objet):
    """
    Renvoie le nombre de points d'utilités d'un objet
    :param objet : (dict) représentant un objet
    :return: (int) le nombre de points d'utilités
    """
    return objet['pts']

def trier_liste (liste_objets) :
    return sorted(liste_objets, key=points_d_utilites, reverse=True)
```

6. Ouvrir le fichier glouton.py et tester les lignes suivantes :

```
>>> points_d_utilites(LISTE_OBJETS[0])
???
>>>  liste_objets = trier_liste (LISTE_OBJETS)
>>> liste_objets
???
```

7. Quel est le rôle de la fonction trier_liste ?

8. Ecrire la docstring de la fonction trier_liste.

```
```

9. Quel est l'avantage de trier la liste dans l'ordre décroissant de points d'utilités ?

```
```

10. Implémenter à partir du pseudo code suivant la fonction choix_objets qui répond à la spécification donnée dans le fichier fourni.

```
Trier la liste d'objets disponibles
Initialiser la solution à tableau vide
Initialiser i à 0
Initialiser le score à 0
Tant que i est plus petit que la longueur de la liste d'objets et que la surface
disponible est non nulle:
    Stocker dans objet le dictionnaire stocké dans la liste à l'indice i
    Si la surface disponible moins la taille de l'objet est str plus grand que
0:
        Ajouter le nom de l'objet à la solution
        Soustraire la taille de l'objet à la surface disponible
        Ajouter le nombre de points d'utilités au score total
    Incrémenter i de 1
Renvoyer la solution, le score total
```

11. Tester votre fonction avec comme surface 25 puis 36. Vérifier alors les résultats trouvés aux questions 2 et 3.

Nous souhaitons aménager une seconde pièce de notre maison ayant pour dimensions **4x5 cases**, nous avons décidé de faire une cuisine, pour cela nous possédons la liste d'objets suivante :

Noms	Dimensions	Nombres de cases	Points d'utilité
Four	1x2	2	20
Placard	1x2	2	20
Lave-vaisselle	1x2	2	10
Réfrigérateur	1x2	2	30
Plaque électrique	1x2	2	30
Table	3x3	9	50
Evier	2x2	4	30
Micro - ondes	1x2	2	30

Noms	Dimensions	Nombres de cases	Points d'utilité
Cafetière	1x2	2	20
Plante	1x2	2	5

12. Ecrire la liste de dictionnaires correspondante à nos nouveaux objets.

13. Ecrire la liste de dictionnaires correspondante à nos nouveaux objets.

Résultat obtenu :

14. Cette solution est-elle la meilleure ? N'y-a-t-il pas de combinaisons maximisant le score d'utilité autre que celle obtenue avec l'algorithme ?

Nota bene :

Dans ce TP, nous avons réfléchi et mis en œuvre une stratégie afin d'optimiser le nombre de points d'utilités. Nous avons convenu de choisir à chaque fois l'objet possédant le plus de points d'utilités pouvant être placé dans la surface disponible. Cette stratégie met en œuvre une succession de choix optimaux locaux sur des problèmes de plus en plus petits. Le nom officiel de la famille d'algorithmes utilisant cette stratégie est **Algorithme glouton**.

Définition : Optimiser un problème, c'est déterminer la solution répondant à une caractéristique spécifique. Par exemple, déterminer le minimum ou le maximum d'une fonction est un problème d'optimisation. Optimiser le problème de ce TP, est trouver l'ensemble de objets visant à maximiser le nombre de points d'utilités pour une surface donnée.

JHermilier