Enoncé

Rendez-vous sur https://classroom.github.com/a/hx-LhCIB et téléchargez votre projet. Remplissez correctement tous les bouts de code et répondez correctement à toutes les questions. Les fonctions correctes ne peuvent vous assurer une note parfaite. La qualité de votre code, performance et lisibilité sont également pris en compte !

Pour finir, n'oubliez pas de *pusher* votre code sur votre *repository* git – vous avez jusqu'au 13 octobre 2019 à 23h59.

Exercice 1

exercice.py

La fonction `multiple_de` doit nous permettre de savoir si un nombre est un multiple d'un autre. Par exemple :
Pour un appel `multiple_de(2, 10)`, la réponse doit être `true`.
Pour un appel `multiple_de(3, 20)`, la réponse doit être `false`.

Vous pouvez utiliser l'opérateur "modulo".
https://blog.tecladocode.com/pythons-modulo-operator-and-floor-division/

questions.txt

Répondez aux questions dans le fichier questions.txt!

Exercice 2

exercice.py

Voici une classe `ArrayList` qui doit gérer un tableau de mots.
La fonction `__init__` correspond au constructeur de la classe.
Le mot clé `self` indique qu'une fonction se réfère à une instance et non pas à une classe.
Pour comparaison, ne pas mettre `self` est l'équivalent d'avoir `static` dans une méthode Java.

questions.txt

Après avoir analysé ce code, répondez aux questions.

Exercice 3

exercice.py

Voici une classe `NumberList` qui génère 15 numéros aléatoires entre 0 et 10.
Vous devez coder la partie manquante dans la fonction `max` qui doit retourner **l'indice** du plus grand numéro de la liste.
S'il existe des doublons de ce même numéro, retourner le plus petit indice.

Exercice 4

Partie 1 – Code

exercice.py

Etant donné une équation à 3 inconnues, donner toutes les solutions possibles pour lesquelles a, b et c sont plus petits que n et dont le résultat est $= 100$.

Imaginons cette équation :

$$5x + 10y + 20z = 100$$

Pour résoudre cette équation, votre fonction sera appelée de la façon suivante :

```
equation(5, 10, 20, 10)
```

Où 5 est le poids du x, 10 le poids du y, 20 le poids du z et ensuite le numéro 10 correspond à la limite des valeurs qui résolvent l'équation. C'est-à-dire, que x, y et z ne peuvent pas être supérieurs à 10. Le premier résultat retourné sera :

$$5 \cdot 0 + 10 \cdot 0 + 20 \cdot 5 = 100$$

Votre fonction doit retourner une liste de dictionnaires – n'utilisez que des nombres entiers.

```
[{'x': 0, 'y': 0, 'z':5 }, ... ]
```

Partie 2 – Questions

questions.txt

Répondez aux questions dans le fichier questions.txt.

Partie 3 - Tests de performance

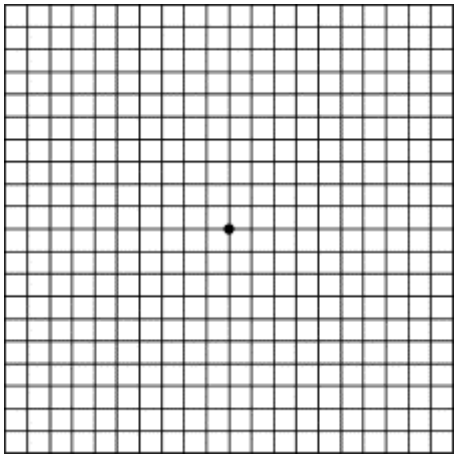
performance.py

En moyenne, quelle est la durée d'exécution de cet algorithme pour $n = 10$.
Démontrez-le en utilisant du code Python.
Et pour $n = 100$?
Et pour finir, pour $n = 1000$?

A l'aide de *matplotlib*, créez un graphique affichant la différence de performance entre les plusieurs n. (Voir le fichier exemple.py pour un exemple d'un graphique)
Attention: Afin d'avoir un calcul plus précis, vous devez le faire beaucoup de fois (10000 p.e.) et utiliser sa moyenne.
BONUS: Expliquez pourquoi nous devons utiliser une moyenne.

Exercice 5

Dans cet exercice, une matrice de taille $n \times m$ contient un point p :



A priori, vous ne connaissez ni la taille de la matrice, ni la position du point p. Votre objectif est de le trouver ce point p.
Pour information, l'intersection en haut à gauche est considérée comme le point $(0,0)$.

Partie 1

exercice_1.py

Codez une version naïve de l'algorithme de recherche. C'est-à-dire, tous les points doivent être vérifiés avant de le retourner.

Partie 2

exercice_2.py

Ici, améliorez votre code et retournez le point p une fois trouvé.

Partie 3

questions.txt

Répondez aux questions dans le fichier questions.txt.

Partie 4

performance.py

Vérifiez vos réponses au moyen d'un test de performance.

Créez 1000 matrices de taille 10×10, 1000 matrices de taille 100×100 et 1000 matrices de taille 1000×1000.

Ensuite, créez un graphique permettant de comparer la performance moyenne des deux algorithmes.

Attention: Afin d'avoir une comparaison correcte, vous devez utiliser les mêmes matrices pour les tests de la version naïve et la version améliorée!

Exercice 6

Partie 1

exercice_1.py

Sans utiliser l'opérateur ****** ni la fonction **math.pow**, créez une fonction permettant de calculer a^n

Partie 2

exercice_2.py

Toujours sans utiliser l'opérateur ****** ni la fonction **math.pow**, réécrivez votre fonction mais avec une complexité de $log(n)$ si n est une puissance de 2.

Partie 3

questions.txt

Répondez aux questions dans le fichier questions.txt.

Exercice 7

Dans cet exercice, c'est à vous de décider comment votre code sera structuré !

L'objectif est d'utiliser des **Piles**, des **Files** et des **Tableaux** pour développer un algorithme permettant de savoir si un code est correct. Ici, nous vous demandons uniquement de vérifier les parenthèses ( ), les accolades { } et les crochets [ ]. Voici des exemples de codes corrects :

{({})} – Correct.

{([()]}} – Faux, il manque une parenthèse fermante et il y a une accolade fermante de trop.

Vous devez charger le code source de chaque fichier, l'analyser et afficher à l'utilisateur quels fichiers sont corrects et lesquels possèdent des erreurs.

Plusieurs types de langages sont utilisés : Java (.java), C# (.cs) et PHP (.php). Votre code doit pouvoir supporter tous ces langages ci-dessus.

Voici certaines classes et leurs responsabilités :

CharsStorage : connaître les caractères ouvrants en fermants dans un fichier donné

FileAnalyser : connaître l'algorithme qui permet de déterminer si le code est correct

Loader : gérer le chargement des fichiers

Queue : gérer une queue

Stack : gérer un stack

Bien évidemment, vous devez coder vous-même la Queue et le Stack. Des implémentations existent déjà dans Python, mais il est interdit de les utiliser dans ce TP.

Votre output doit correspondre à ceci :

```
Voici les fichiers qui contiennent des erreurs :
Dechet
DechetMisMatchException
Ecole
Form1
Form3
HomeController
ListeDechets
UserController

Voici les fichiers qui sont corrects :
Controller
DechetAutre
DechetCategorie
DechetCompost
DechetMetal
DechetPET
DechetPapier
DechetVerre
DechetsDao
FileStructureException
Form2
Form4
FrmTri
Personne
RestaurantController
UnknownDechetTypeException
```

L'extension du fichier ne doit pas être affichée et l'ordre des fichiers doit être trié par ordre alphabétique.