Formation Python, perfectionnement
Sujet des travaux pratiques

Matthieu Falce

Janvier 2023

1 Manipulation de la syntaxe python

1.1 *Fizz Buzz*

Concepts : 1) variables 2) boucles 3) conditions

Implémentez le code correspondant à l'algorithme suivant :

Pour les nombres de 1 à 100,
si le nombre est divisible par 3, écrivez Fizz ;
s'il est divisible par 5, écrivez Buzz ;
s'il est divisible par les deux, écrivez FizzBuzz ;
sinon écrivez le nombre.

1.2 Plus ou moins

Concepts : 1) variables 2) boucles 3) conditions 4) IO 5) exceptions

Vous allez coder un jeu simple pour enfant.

Étapes :

1. l'ordinateur choisit un nombre et l'utilisateur doit le deviner
2. l'ordinateur demande à l'utilisateur d'entrer un nombre
3. l'ordinateur ne peut dire que *c'est plus* ou *c'est moins*
4. Le jeu s'arrête quand l'utilisateur a trouvé le bon nombre.

Questions :

1. Comment pouvez-vous gérer les entrées invalides ?

1.3 Slices

Concepts : 1) structures de données 2) découpages 3) gestion des exceptions

Nous allons réimplémenter plusieurs commandes *shell* permettant d'afficher le contenu d'un fichier de différentes façons.

Questions :

1. implémenter la commande `unix cat` (affiche le contenu dans la console)
2. implémenter la commande `unix tac` (affiche le contenu à l'envers)
3. implémenter la commande `unix head` (affiche les N premières lignes du fichier)
4. implémenter la commande `unix tail` (affiche les N dernières lignes du fichier)
5. n'afficher qu'une ligne sur 3 entre la 5^{e} ligne et la 10^{e} ligne avant la fin
6. qu'en conclure sur les performances ? Comment faire avec de gros gros fichiers ?

1.4 Magic 8 ball

Concepts : 1) aléatoire 2) types 3) manipulation de fichiers 4) manipulation de chaînes

Vous n'arrivez pas à vous décider. Un programme peut vous aider à vous guider dans la vie.

Étapes :

1. chargez les phrases du fichier `exercices/python/magic_8_ball/media/phrases_magic_8_ball.txt`
2. affichez en une au hasard

1.5 Comparaison de textes

Concepts : 1) manipulation de fichiers sur le réseau 2) familiarisation avec les types de bases (ensembles, dictionnaires) 3) découverte de la bibliothèque standard

Vous allez récupérer 2 livres du projet Gutemberg.

— `https://www.gutenberg.org/cache/epub/51804/pg51804.txt`

— `https://www.gutenberg.org/files/53311/53311-0.txt`

Questions :

1. comment télécharger depuis internet ?
2. quels sont les mots qui apparaissent dans le texte ?
3. lister les mots qui n'apparaissent qu'une fois dans chaque texte. Si l'on regroupe les textes ?
4. lister les mots communs aux deux textes
5. quels sont les 10 mots qui apparaissent le plus ?

Réfléchissez à la complexité algorithmique ($\mathcal{O}(n)$, $\mathcal{O}(n^2)$, $\mathcal{O}(\log(n))$) de votre solution.

1.6 C'est loooong

Concepts : 1) fonctions d'ordre supérieur 2) décorateur 3) gestionnaire de contexte (context manager)

Mesurez de façon générique le temps d'exécution d'une fonction.
La modification doit être la moins invasive possible.

Étapes :

1. codez une fonction qui fait une pause de 1s
2. créez un décorateur de mesure du temps
3. créez un context manager de mesure du temps

Questions :

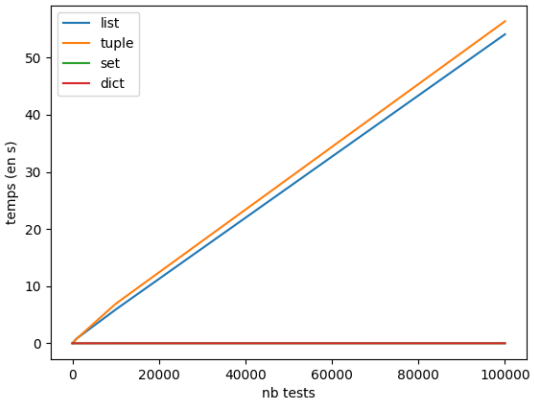
1. comment faire une pause dans un programme en Python ?
2. comment mesurer le temps d'execution d'un bout de programme en Python ?
3. comment faire des analyses statistiques plus poussées sur le temps d'exécution ?
4. Des solutions inclues dans python existent-t-elles ?

1.7 Analyse de complexité

Concepts : 1) structures de données 2) complexité

Nous allons analyser la complexité algorithmique des opérations `in` (*est présent*) pour divers conteneurs de la bibliothèque standard (`list`, `tuple`, `dict`, `set`).
Pour cela, il faut mesurer un temps d'exécution en fonction de la taille des paramètres en entrée.

Un graphique d'analyse de complexité est présent à droite



Questions :

1. quel protocole d'analyse utiliser ?
2. comment l'implémenter ?
3. qu'en conclure ? (qu'en dit la documentation)
4. (pro) comment réaliser ce graphique ?

2 Fonctions avancées

Cette section comporte des exercices qui mettent en pratique les manipulations avancées de fonctions

2.1 Miam

Concepts : 1) décorateurs 2) décorateurs multiples 3) décorateurs paramétrés

Codez un décorateur permettant de dessiner un hamburger.

On souhaite quelque chose de cette forme :

```
def ingredient_1(...):
    print("l'ingrédient 1")

def ingredient_2(...):
    print("l'ingrédient 2")

@ingredient_1
@ingredient_2
def sandwich(viande):
    print(viande)
```

Étapes :

1. la fonction a décorer prend le type de viande (ou végétal...)
2. écrire un décorateur pour les ingrédients (tomate, fromage, salade), que l'on peut placer dessus ou dessous
3. écrire un décorateur pour le pain
4. créer un sandwich (de haut en bas) : pain, salade, tomate, steak, fromage, salade, pain

Questions :

1. dans quel ordre les décorateurs sont-ils appliqués ?
2. peut-on avoir le même comportement avec un seul décorateur paramétré ? Comment l'implémenter ?

2.2 Déprécié

Concepts : 1) fonctions d'ordre supérieur 2) décorateur 3) introspection

Créer un décorateur qui permette de marquer une fonction ou une classe comme dépréciée.

Étapes :

1. on veut marquer une fonction ou une classe
2. on veut savoir le nom, le numéro de ligne et le fichier de ce que l'on a marqué
3. on veut avoir la liste de ce que l'on a marqué

Questions :

1. à quel endroit du décorateur doit-on marquer les fonctions ?

2.3 *Fizz Buzz* is back

Concepts : 1) variables, boucles, conditions 2) paresse

Implémentez le code correspondant à l'algorithme suivant :

Pour les nombres de 1 à 100,
si le nombre est divisible par 3, écrivez Fizz ;
s'il est divisible par 5, écrivez Buzz ;
s'il est divisible par les deux, écrivez FizzBuzz ;
sinon écrivez le nombre.

Étapes :

1. utiliser un générateur
2. un itérateur
3. des générateurs chaînés

Questions :

1. cela est-il pertinent ?

2.4 Pipeline grep

Concepts : 1) pipeline 2) consommateurs

Implémentez la commande `Unix grep`.

Étapes :

1. générez un flux de données simulant un serveur web : infini avec des pauses entre chaque événements (en utilisant le fichier `media/log_apache_exemple.txt`)
2. rajouter un filtre sur les erreurs 404
3. rajouter un filtre calculant le poids total des images envoyées

Questions :

1. quel type de pipeline utiliser ?

2.5 Pipeline filtres

Concepts : 1) générateur 2) pipeline 3) `design pattern`

Réimplémentez un pipeline comme vu en cours.

Étapes :

1. pipeline des nombres divisibles par 3 et palindromes
2. rajouter le filtre suivant : nombres qui contiennent un 2 ou un 7
3. rajouter le filtre suivant : nombres qui commencent par un 1

Questions :

1. comment organiser le code pour rajouter facilement de nouveaux filtres ?

2.6 ItertoolsBis

Concepts : 1) itertools 2) itérateurs

Vous aller recoder des fonctions de la bibliothèque `itertools`

Étapes :

1. recoder `chain`
2. recoder `cycle`

2.7 Pareisseux

Concepts : 1) paresse 2) `itertools`

Regardez le module `itertools` [1].

Questions :

1. quel peut-être l'intérêt pour recoder les outils UNIX ?

2.8 C'est loooong

Concepts : 1) fonctions d'ordre supérieur 2) décorateur 3) gestionnaire de context (context manager) 4) gestionnaire de context en utilisant le décorateur

Mesurez de façon générique le temps d'exécution d'une fonction.
La modification doit être la moins invasive possible.

3 Programmation orientée objet

3.1 Formation

Dans cette section vous allez améliorer le même code petit à petit afin d'avoir une application permettant de gérer les formations.

1. `https://docs.python.org/3.7/library/itertools.html`

3.1.1 Tout commence par des humains

Concepts : 1) création de classes 2) méthodes et attributs 3) méthodes magiques

Créez une classe `Personne`. Elle va contenir les informations d'une personne (en gros les informations d'une carte d'identité).
Une personne doit pouvoir dire si elle est majeure. Attention, la limite d'âge de la majorité peut changer et doit donc être dynamique.

```
p1 = Personne(naissance=1990, nom="Matthieu")
p2 = Personne(naissance=2015, nom="Paul")

assert p1.est_majeur()
assert not p2.est_majeur()

Personne.AGE_MAJORITE =40

print(p1.est_majeur() == False)
print(p2.est_majeur() == False)
```

La personne doit pouvoir se présenter en faisant :

```
p1 = Personne(naissance=annee_naissance, nom=nom)
print(p1)
```

3.1.2 Attributs, Getters, Setters

Concepts : 1) `properties`

L'age d'un `Eleve` ne se recalcule pas. Comment régler ce problème de la façon la moins intrusive possible ?
On veut pouvoir faire

```
p1 = Personne("1", 1950)
print(p1.age == 68)

p1.age = 10
print(p1.annee_naissance == 2008)

p1.annee_naissance = 0
print(p1.age == 2018)
```

3.1.3 Puis des élèves et des profs

Concepts : 1) héritage 2) surcharge

Dessinez les diagrammes de classes correspondant à ce qui est demandé.
Créez `Eleve` et `Prof` qui sont des `Personnes`.
Un `Prof` doit pouvoir dire si un élève a payé ses frais de scolarité.

3.1.4 Et enfin, une formation

Concepts : 1) composition

Une `Formation`, c'est des élèves et des profs en même temps.
Un élève doit être majeur et avoir payé ses frais de scolarité pour participer. Modifiez le prof pour qu'il sache si un élève peut s'inscrire ou pas.

3.2 Convertisseur de température

Concepts : 1) `property` 2) accès aux attributs

Comment implémenter un convertisseur de température Celsius / Fahrenheit ?
Pour rappel :

$$T_{[^\circ C]} = (T_{[^\circ F]} - 32) * 5/9$$
$$T_{[^\circ F]} = T_{[^\circ C]} * 9/5 + 32$$

Questions :

1. essayer de gérer 2 attributs simultanément. Quels problèmes rencontre-t-on ?
2. essayer la même chose en utilisant les `properties`. Quel est l'intérêt de cette solution ?

4 POO avancée

Cette section comporte des exercices qui mettent en pratique les manipulations avancées d'objets

4.1 Pathlib

Concepts : 1) héritage 2) concept de `self`

Il existe la bibliothèque `pathlib` en python. Elle permet de manipuler des arborescences de fichiers sous forme d'objets. Sa documentation est disponible ici : https ://docs.python.org/fr/3/library/pathlib.html
Votre mission est de coder une class `Path` qui simule le même comportement : `print(Path("") / "etc" / "nginx")` doit afficher "/etc/nginx"

Étapes :

1. créer la classe qui permet de faire ce qui est demandé
2. trouver un moyen de faire une classe multi-plateforme (le séparateur d'arborescence sous Unix n'est pas le même que sous Windows)

4.2 Properties

Concepts : 1) properties 2) décorateur 3) descripteur

Implémenter le décorateur property avec un descripteur en pur python.

Étapes :

1. mettre en place le `getter`
2. mettre en place le `setter`

4.3 La classe, la méta-classe

Concepts : 1) métaclasses

Pour bien comprendre les métaclasses.

Questions :

1. quelle est la différence entre surcharger le ___new___ d'une classe et d'une métaclasse ?
2. pareil pour le ___call___
3. sur quels éléments ces méthodes s'appliquent ?

4.4 Singleton

Concepts : 1) `design pattern` 2) métaclasses

Implémentez le design pattern singleton en utilisant une métaclasse.

Questions :

1. quelle fonction de la classe veut-on modifier ?
2. qu'elle évolution avec le code du cours ?
3. quelle méthode préférez vous ? Pourquoi ?

5 Modules

5.1 Bases de données

Concepts : 1) manipulation de la DB api

Vous allez réaliser une application en ligne de commande permettant de stocker les notes d'un utilisateur dans une base de données.

Questions :

1. quelles sont les informations à stocker ?
2. comment effectuer des recherches spécifiques ?

5.2 Expressions régulières

Tous, trouvez les tous
Vous allez travailler sur le fichier : `medias/modules/pythonVersions.txt`

Questions :

1. quelles sont les versions de python présentes dans le fichier ?
2. comment peut on savoir laquelle est la plus présente ?

Cherchez / trouvez
Concepts : 1) syntaxe des expressions régulières 2) manipulation des expressions régulières

Trouvez les occurrences et positions des mots "chat" dans le texte `medias/modules/texteRegex.txt` .

Questions :

1. Comment modifier l'expression pour reconnaître aussi le mot "cat".
2. Comment modifier l'expression pour ne pas reconnaître les mots contenant le mot "chat ou "cat (cathédrale par exemple) ?
3. est-ce que les expressions régulières sont plus adaptées que les outils de manipulation de chaînes inclus en python dans ce cas ?

Identifiants
Nous avons des chaines de cette forme ci :

— `id/233b6a88-fd22-4e28-a636-6ebea175dc34/review`
— `id/f28b9b71-ba38-4/review`
— `id/40312d67-85c2-4/review`
— `id/48a8da09-1ae8-474/review`
— `id/523c5cdf-d830-4b97-b/review`
— `id/124defcd-e4d1-4c2a/review`

Questions :

1. Voyez-vous un motif dans ces chaînes ?
2. Comment extraire l'information stockée ?

C'est valide

Forme canonique d'une adresse email : `identifiant@domaine.tld`

Questions :

1. Comment peut-on imaginer valider une adresse e-mail ?
2. Imaginez qu'un utilisateur entre une adresse, comment vérifier si elle correspond à la forme canonique ?
3. Comment se servir de cela pour valider les mails présents dans `medias/modules/listingMail.json` ?

6 Parallélisme et performances

On va regarder ici différentes méthodes permettant d'améliorer les performances d'une application.

6.1 Mise en cache

Concepts : 1) décorateur 2) performances

Pourquoi calculer quand on peut se souvenir.

Étapes :

1. calculez la suite de fibonacci de façon récursive
2. codez le décorateur qui va stocker et renvoyer le résultat de la fonction si le calcul à déjà été fait
3. calculez les différences de performances entre les deux solutions

Pour rappel, voici la définition de la suite de Fibonnaci :

$$\begin{cases} f_0 = 0 \\ f_1 = 1 \\ f_n = f_{n-1} + f_{n-2} \end{cases}$$

Questions :

1. quelles sont les limites de cette solution ?
2. comparez avec la fonction `lru_cache` de la bibliothèque standard.

6.2 Travailleurs !

Concepts : 1) programmation concurrente 2) performances 3) `beautiful soup`

Compter les mots les plus utilisés sur 10 articles de wikipedia à choisir.

Étapes :

1. choisir 10 articles de wikipedia
2. télécharger une page et compter les mots utilisés
3. regrouper les mots pour tous les articles
4. lancer le code dans un worker `thread`
5. lancer le code dans un worker `multiprocess`

Questions :

1. que peut-on penser des performances à priori entre les méthodes utilisées ?
2. comment faire pour limiter le nombre de requêtes au site par minute ?

7 Intégration Python / C

7.1 Intégration code natif

Concepts : 1) échange de données entre les deux langages 2) `ctypes` 3) `cython`

Appelez du code C en python.

Étapes :

1. créer une fonction en `C` réalisant une addition de deux int et retournant le résultat
2. appeler cette fonction depuis python en utilisant
   - `ctypes`
   - `cython`
3. appelez maintenant une fonction concaténant deux *strings* en `C`

Questions :

1. quel impact a chaque technique sur les performances ?
2. quel avantage de cython par rapport à ctypes

7.2 On embarque

Concepts : 1) échange de données entre les deux langages 2) `Python C-API`

Appelez du code Python en C.

Étapes :

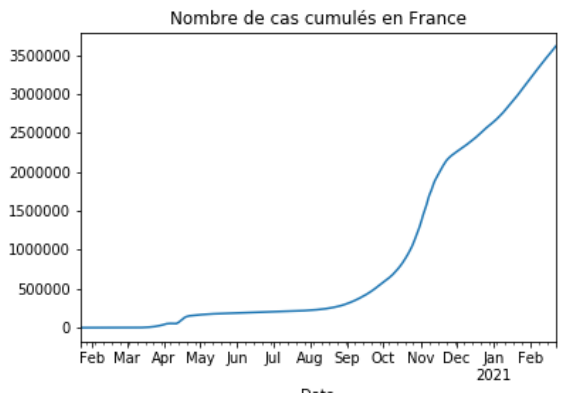
1. créer un fichier python définissant une fonction manipulant des nombres
2. appeler cette fonction depuis un programme C

8 Utilisation en *datascience*

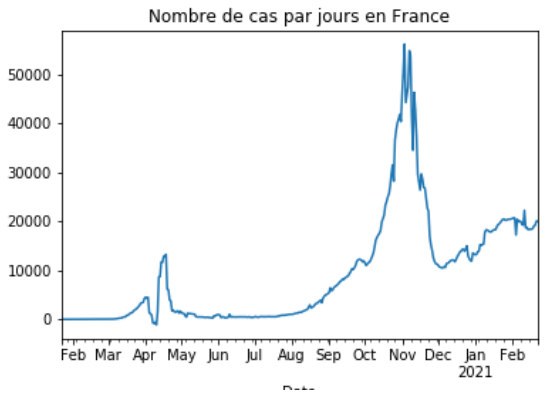
8.1 Analyse Covid

Concepts : 1) `pandas` 2) `manipulation de séries temporelles` 3) `création de graphiques cartographiques` 4) `fusion de dataframes`

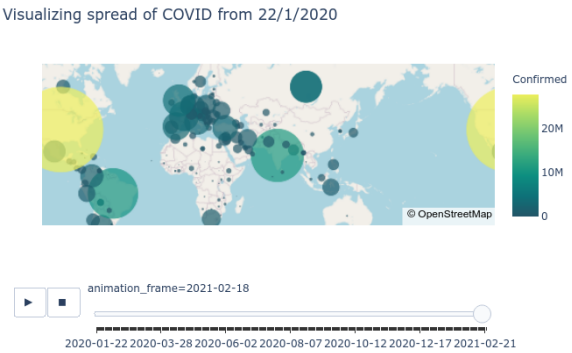
Nous allons analyser des données de l'épidémie de Covid-19.
Les analyses sont libres, vous trouverez quelques exemples ci-dessous (réalisation de visualisation temporelle et géographique de la situation).



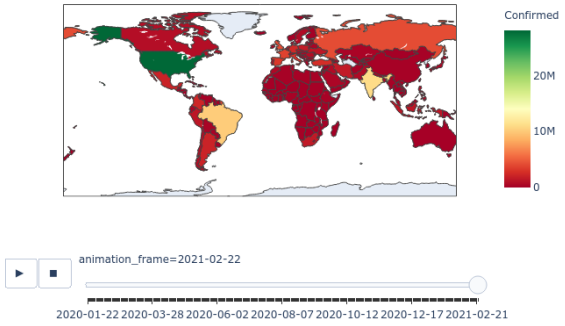
Cas cumulés de Covid en France



Cas journaliers de Covid en France



Carte scatter des cas de covid dans le monde



Choropleth des cas de covid dans le monde

Étapes :

1. les données de covid sont ici : `https://raw.githubusercontent.com/datasets/covid-19/master/data/countries-aggregated.csv`
2. il faut effectuer une moyenne glissante sur 7 jours (par pays) pour fiabiliser les données (lutter contre *l'effet weekend*)
3. pour obtenir les codes ISO3 des pays (nécessaire pour faire les cartes), la bibliothèque `country_converter` peut-être utilisée
4. les coordonnées des barycentres des pays sont ici : `https://gist.githubusercontent.com/tadast/8827699/raw/f5cac3d42d16b78348610fc4ec301e9234f82821/countries_codes_and_coordinates.csv`