

Formation Python avancé ING Luxembourg

Sujet des travaux pratiques

Matthieu Falce

Janvier 2024

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 Gutenberg.

- <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'exécution d'un bout de programme en Python ?
3. comment faire des analyses statistiques plus poussées sur le temps d'exécution ?
4. Des solutions incluses dans python existent-elles ?

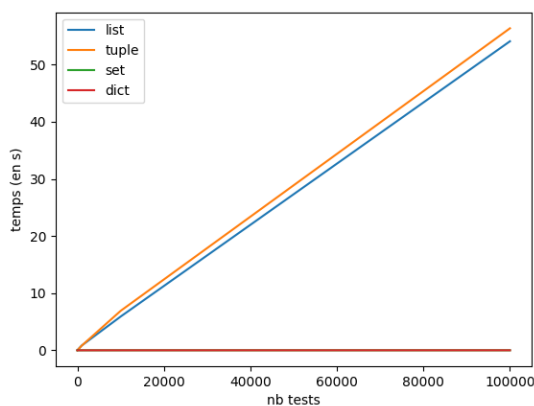
1.7 Analyse de complexité

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 Programmation orientée objet

2.1 Formation

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

2.1.1 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)
```

2.1.2 Attributs, Getters, Setters

Concepts : 1) `properties`

L'âge 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)
```

2.1.3 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.

2.1.4 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.

2.2 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 ?

3 Modules

3.1 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 ?

3.2 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` ?

4 Flask

4.1 Mini projet

Concepts : 1) création d’une application flask 2) utilisation de templates 3) creation de vues et leur routage 4) manipulation de formulaires

Étapes :

1. créer le projet flask

2. créer une vue qui renvoie "Hello, World!" sur la route principale (l'index : "/")
3. créer une vue qui accepte un paramètre dans l'URL (par exemple, un nom d'utilisateur) et renvoie un message personnalisé (stockez les informations dans une variable python, pas besoin de base de données) : que ce passe-t-il si le nom d'utilisateur n'est pas présent ?
4. modifier cette vue pour renvoyer une page HTML à partir un template
5. créer une page avec un formulaire HTML et une vue qui traite les données du formulaire en utilisant Flask-WTF (les affiche dans la console)
6. créer des gestionnaires d'erreurs personnalisés pour les erreurs 404 et 500 (par exemple, vous ne réagissez qu'aux utilisateurs présents dans la variable des noms utilisateurs)