Formation python Orienté Objet et interfaces graphiques
Sujet des travaux pratiques

Matthieu Falce

Avril 2022

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 :

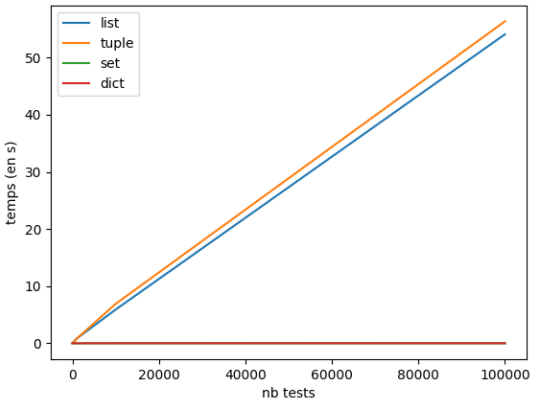
1. comment faire une pause dans un programme en Python ?
2. comment mesurer le temps d'execution d'un bout de programme en Python ?
3. comment faire des analyses statistiques plus poussées sur le temps d'exécution ?
4. Des solutions inclues dans python existent-t-elles ?

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 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")

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

Personne.AGE_MAJORITE = 30

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'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)
```

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 :

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

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 Fils rouges

Cette section comporte des exercices qui sont transverses à plusieurs parties ou qui nécessitent d’avoir une vision plus globale de l’architecture d’un projet python.

4.1 On reprend tout

Concepts : 1) vision globale d'un projet python

Nous allons voir comment faire un petit projet de A à Z. Il s'agit de code une solution permettant de gérer des formations. La liste des élèves et des professeurs sera stockée dans une base de données et affichée dans une interface graphique.

Étapes :

1. adapter le code de la section 3.1 (page 4) pour enregistrer les participants dans une base de données. Quel est l'intérêt ?
2. modifier l'interface développée dans la section 5.5 (page 7) permettant de manipuler des données de personnes
3. créer une documentation
4. rajouter des tests unitaires

Quelles fonctionnalités peut-on rajouter ? Comment peut-on faire ?

4.2 Un géant aux pieds d'argiles

Concepts : 1) vision globale d'un projet python 2) mettre en place des stratégies de débug 3) gestion de *legacy code*

Vous héritez d'un projet python d'une haute importance stratégique. Il se situe dans le fichier `medias/filRouge/shifumi.py` . Bon, il ne fonctionne pas, mais vous savez, les mauvais recrutements ça arrive...
Le but du projet est de simuler un *pierre-papier-ciseaux*. Vous devez le faire marcher sans tout recoder.

Étapes :

1. regardez le code, essayez de le lancer
2. faire le ménage au niveau stylistique
3. régler les problèmes empêchant le code de se lancer
4. régler les problème de logique

N'hésitez pas à utiliser tous les outils que vous avez à disposition (débuggeur, linters, outils de formatage automatiques, outils de refactoring, tests unitaires)

5 Interfaces graphiques avec Tkinter

5.1 Des boutons partouts

Concepts : 1) interfaces graphiques 2) manipulation de `layout` 3) évaluation de code

Vous aller implémenter une interface graphique pour une calculatrice.

Questions :

1. comment peut-on obtenir le résultat d'un calcul sous forme de chaîne de caractères (`2*4+6` par exemple) ?
2. quel choix de `layout` allez vous effectuer ?
3. comment calculer le résultat d'une opération rentrée par l'utilisateur ?

5.2 C'est pas clair tout ça

Concepts : 1) interfaces graphiques 2) notion de design / UX

Vous aller chercher à implémenter la *pire interface possible* pour demander un numéro de téléphone à un utilisateur.

Questions :

1. quels *inputs* peut-on utiliser ? Justifier vos choix.
2. quelle interface doit-on utiliser pour faciliter la tâche à l'utilisateur ?

5.3 Pitichat

Concepts : 1) boucle d'événement 2) threading 3) téléchargement de ressources internet 4) affichage d'images

Nous allons voir comment faire un petit projet de A à Z. Il s'agit de code une solution permettant de gérer des formations. La liste des élèves et des professeurs sera stockée dans une base de données et affichée dans une interface graphique. Vous aller prototyper la nouvelle interface graphique de `kawai as a service`. Nous voulons une interface permettant d'afficher un animal mignon (récupérée sur `https://loremflickr.com/`) avec un bouton qui permette de changer d'animal.

Étapes :

1. télécharger une image depuis loremflicker (utiliser `requests`)
2. afficher cette image depuis l'interface graphique
3. rajouter un bouton permettant de changer d'image (en téléchargeant une nouvelle image). Quel problème rencontre-t-on dans l'interface ?

Questions :

1. pourquoi ne peut-on pas intéragir avec l'interface pendant qu'une image s'affiche ?
2. peut-on reproduire le problème simplement ?
3. comment régler ce problème ?

5.4 Artistes

Concepts : 1) interfaces graphiques 2) manipulation de dessins 3) gestion des événements souris 4) gestion des menus

Implémentez un logiciel de dessin (un peu comme `MS Paint`).

Questions :

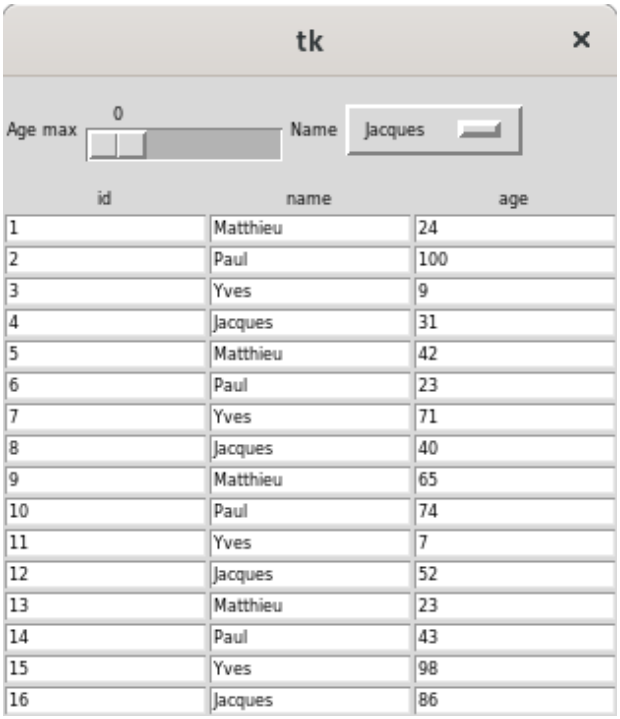
1. quel type de widget utiliser ?
2. comment changer de couleurs ?
3. comment peut-on sauvegarder un dessin ?
4. comment intégrer le changement de couleurs dans le menu ?

5.5 Je table sur ça

Concepts : 1) interfaces graphiques 2) manipulation de données 3) mise à jour des fenêtres

Implémentez un logiciel de visualisation de données tabulaire.

Vous gérez un service RH et vous voulez trier vos employés par nom ou par age. Voilà le genre de logiciel que vous devez obtenir



Étapes :

1. créez une interface permettant de lister les informations dans des cellules (comme dans un tableur)
2. rajouter un bouton permettant de filtrer par nom
3. rajouter un bouton permettant de n'afficher que les personnes ayant un âge supérieur à celui demandé

Questions :

1. comment lier les informations entre les boutons et le tableur ?
2. comment éviter des artéfacts d'affichage lors de la modification ?

6 Écosystème scientifique

6.1 L'addition s'il vous plaît

Concepts : 1) `broadcasting` 2) `reshaping`

Comment fonctionne l'addition ou la multiplication de vecteurs / matrices avec `numpy` ?
Essayez :

1. `np.ones((2, 2)) + 5`
2. `np.ones((2, 2)) + np.arange(4, 8).reshape((2, 2))`
3. `np.ones((2, 2)) + np.arange(4, 6).reshape((1, 2))`
4. `np.ones((2, 2)) + np.arange(4, 6).reshape((2, 1))`
5. `np.arange(4, 6).reshape((2, 1)) + np.arange(4, 6).reshape((1, 2))`

Questions :

1. comment le `broadcast` se comporte avec des tailles différentes ?
2. Comment se comporte la multiplication ?
3. Comment faire des multiplications matricielles ?

6.2 Un vrai peintre

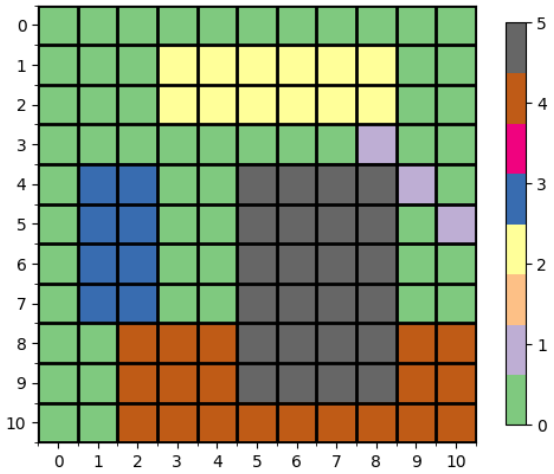
Concepts : 1) `numpy array` 2) `index` 3) `axis`

1. créez un tableau vide de taille 10 avec la valeur 4 en position 5
2. créez un tableau avec les nombres de 13 à 23
3. créez un tableau de taille 854 entre 1 et 2
4. créez un matrice de taille 3x3 avec toutes les valeurs de 1 à 9, transposez là
5. créez la matrice diagonale 3x3
6. créez un tableau 3d aléatoire, de taille (5, 6, 7), calculez les moyennes des couches selon les différents axes
7. créez un échiquier de taille 9x9 (vérifiez avec matplotlib, utilisez `imshow`)
8. normalisez (pour n'avoir que des valeurs entre 0 et 1) le tableau avec les nombres entre 13 et 23
9. créez des tableaux de "grande taille" (200000+ éléments) avec différents types, regardez leur espace mémoire. Essayez avec une grande matrice diagonale et les types de matrices creuses

6.3 Le beau tableau

Concepts : 1) `numpy array` 2) `index`

En utilisant les bonnes `slices`, reproduisez le tableau suivant, en 5 étapes :



6.4 Jouons avec des fonctions

Concepts : 1) vectorisation 2) fonctions universelles

A essayer en python classique et en numpy

Créez un tableau de 842 éléments entre -1 et 1 uniformément répartis.
Appliquer les fonctions suivantes dessus :

1. $f(x) = x^2$
2. $f(x) = cos(x)$
3. $f(x) = \frac{\log(x**2)}{e^{-x}}$

Questions :

1. comment vérifier que les nombres obtenus sont égaux ?
2. comment comparez les performances entre les implémentations ?

6.5 Sinus cardinal

Concepts : 1) grilles 2) `plotting 3d` 3) `heatmap`

Calculez les valeurs du sinus cardinal sur le carré $[-10, 10] * [-10, 10]$
Pour rappel :
$\sin_c(x, y) = \frac{\sin(x^2 + y^2)}{x^2 + y^2}$
Regardez les valeurs avec une heatmap et en 3D

6.6 Rien ne va plus

Concepts : 1) `numpy array` 2) `mask index array` 3) `plotting`

On peut utiliser une méthode de Monte-Carlo pour estimer π.

Étapes :

1. générez 500 nombres répartis aléatoirement sur le carré $[-1, 1] * [-1, 1]$
2. calculez combien de points sont dans le cercle inscrit dans ce carré
3. la probabilité de tomber dans le cercle est égale à $\frac{S_{cercle}}{S_{carre}}$
4. déduisez-en π

Faites la représentation graphique des points que vous avez générés, colorez les selon leur position dans le cercle ou non.

6.7 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.

1. chargez les phrases du fichier `media/phrases_magic_8_ball.txt`
2. affichez en une au hasard l'aide de techniques de `numpy`
3. trouvez plusieurs façons de le faire

Questions :

1. comment tester que la méthode d'échantillonage est vraiment équitable ?

6.8 Marche aléatoire

Concepts : 1) `mask index array` 2) `plotting`

Générez une marche aléatoire sur 1000 points.
Pour rappel, dans une marche aléatoire :

$$\begin{cases} x_{n+1} = x_n + \epsilon_x \\ y_{n+1} = y_n + \epsilon_y \end{cases}$$

Où $\epsilon_{x/y}$ est une variable aléatoire entre -1 et 1.
Pouvez-vous faire la représentation graphique du chemin parcouru, coloré selon la distance.
Pouvez-vous dessiner la distribution des distances ?

6.9 Je me Gausse

Concepts : 1) `matrices` 2) `inversions`

Résolvez le système suivant :

$$\begin{cases} x + 5y = 11 \\ 3x + 2y = 7 \end{cases}$$

Pour rappel, cela revient à résoudre ce calcul :

$$\begin{bmatrix} 1 & 5 \\ 3 & 2 \end{bmatrix} \times \begin{bmatrix} x \\ y \end{bmatrix} = \begin{bmatrix} 11 \\ 7 \end{bmatrix} \implies \begin{bmatrix} x \\ y \end{bmatrix} = \begin{bmatrix} 1 & 5 \\ 3 & 2 \end{bmatrix}^{-1} \times \begin{bmatrix} 11 \\ 7 \end{bmatrix}$$

6.10 Mon champion

Concepts : 1) `pandas` 2) `manipulation de dataframes` 3) `création de graphiques`

On va regarder quels sont les pays qui ont gagnés le plus de médailles aux Jeux Olympiques.

Étapes :

1. téléchargez le CSV suivant dans pandas : `https://raw.githubusercontent.com/realpython/python-data-cleaning/master/Datasets/olympics.csv`
2. changez les noms de colonnes pour avoir les noms suivants :
   - 'Unnamed : 0' ⇒ 'Country'
   - ' ? Summer' ⇒ 'Summer Olympics'
   - '01 !' ⇒ 'Gold'
   - '02 !' ⇒ 'Silver'
   - '03 !' ⇒ 'Bronze'
   - ' ? Winter' ⇒ 'Winter Olympics'
   - '01 !.1' ⇒ 'Gold.1'
   - '02 !.1' ⇒ 'Silver.1'
   - '03 !.1' ⇒ 'Bronze.1'
   - ' ? Games' ⇒ '# Games'
   - '01 !.2' ⇒ 'Gold.2'
   - '02 !.2' ⇒ 'Silver.2'
   - '03 !.2' ⇒ 'Bronze.2'

Comme analyse, vous voulons afficher les pays selon leur nombre de participations :

```
Number of participations: 1
    * British West Indies (BWI) [BWI]
    * Independent Olympic Participants (IOP) [IOP]
Number of participations: 2
    * Australasia (ANZ) [ANZ]
    * Unified Team (EUN) [EUN]
Number of participations: 3
...
```

L'on peut également se demander si les pays suivants font partis du jeu de données :

- France
- Allemagne
- Monaco

La dernière information que nous voulons connaître :

- quels sont les 10 pays avec le plus de médailles ?
- quels sont les pays avec le plus de médailles par participation

6.11 On se mate une série temporelle ?

Concepts : 1) `pandas` 2) `gestion de séries temporelles` 3) `création de graphiques`

On va analyser des données de puissance électrique produite par des énergies renouvelables.

Étapes :

1. téléchargez le CSV suivant dans pandas : `https://data.open-power-system-data.org/time_series/2020-10-06/time_series_30min_singleindex.csv` (attention, il est assez lourd, comment faire pour ne pas le télécharger plusieurs fois ?)
2. transformez le dataframe en série temporelle
3. il y a des données manquantes dans le jeu de données. Combien ? Comment les supprimer (en les remplaçant par une valeur interpolée) ?

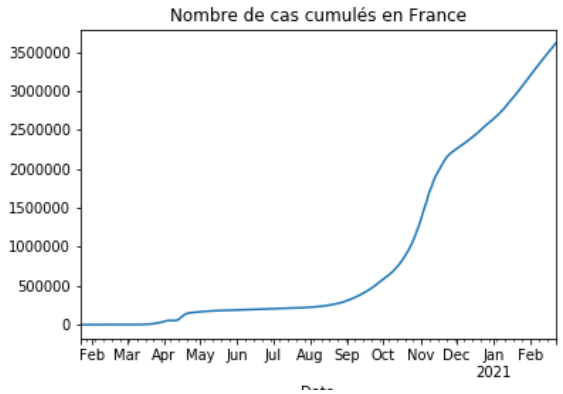
Questions :

1. Quelles analyses peut-on faire sur ces données ?
2. Peut-on voir une saisonnalité dans les données ? Est-elle la même pour le solaire et l'éolien ?

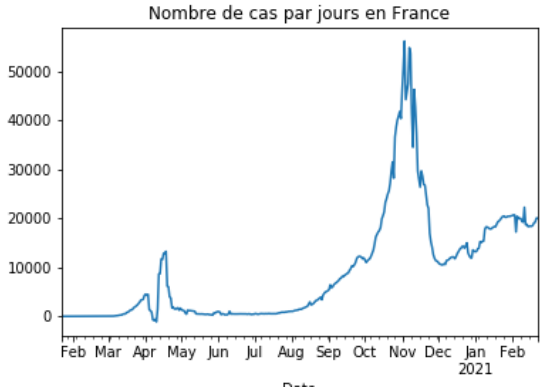
6.12 Analyse Covid

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

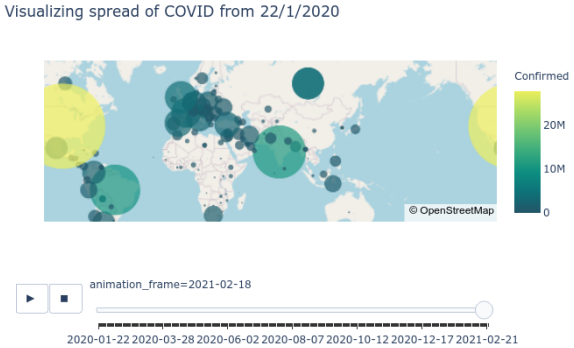
Nous allons analyser des données de l'épidémie de Covid-19.
Les analyses sont libres, vous trouverez quelques exemples ci-dessous (réalisation de visualisation temporelle et géographique de la situation).



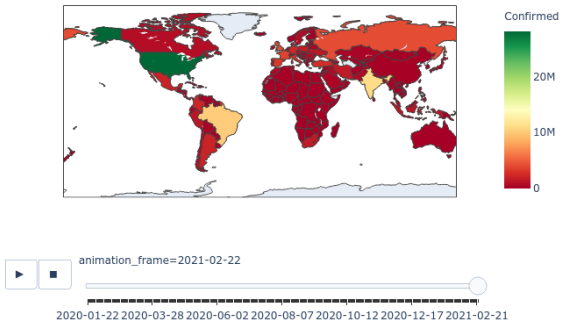
Cas cumulés de Covid en France



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`

6.13 C'est imagé !

Concepts : 1) `numpy array` 2) `manipulation d'images` 3) `détection dans des images`

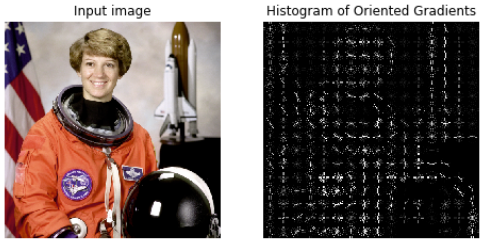
Nous allons tester quelques manipulations classiques d'images.

Étapes :

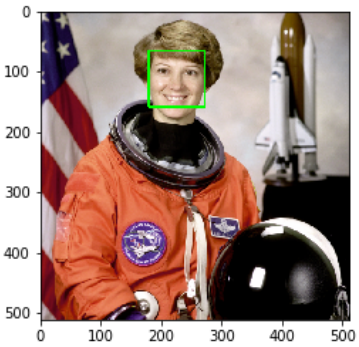
1. à l'aide de skimage appliquez un filtre de Sobel sur une image (détection de contour)
2. binarisez une image en utilisant la méthode d'Otsu pour déterminer le seuil idéal
3. créez un *histogram of gradiants* (HOG) en reprenant le code présent ici : `https://scikit-image.org/docs/dev/auto_examples/features_detection/plot_hog.html`
4. utilisez OpenCV pour effectuer une détection de visage dans une image (utilisez l'image d'astronaute fournie par skimage)
5. effectuez une segmentation d'images en suivant le code suivant : `https://scikit-image.org/docs/dev/auto_examples/segmentation/plot_label.html`. Que pensez-vous du résultat ?

Filtre de Sobel

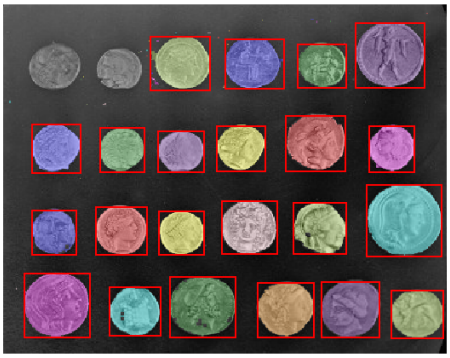
Binarisation avec Otsu



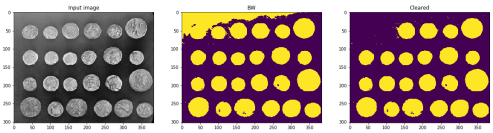
Histogram of gradiants



Détection de visages



Segmentation de pièces



Pièces (intermédiaire)

6.14 Des renards et des lapins

Concepts : 1) vectorisation 2) fonctions universelles

Etudiez une population de lapins et de renards en utilisant le modèle de Lotka-Voltera.
Pour rappel, le système d'équations différentielles est :

$$\frac{dP_{\text{lapins}}}{dt} = \alpha P_{\text{lapins}} - \beta P_{\text{renards}} * P_{\text{lapins}}$$
$$\frac{dP_{\text{renards}}}{dt} = \gamma P_{\text{renards}} - \delta P_{\text{renards}} * P_{\text{lapins}}$$

Avec : $\alpha = 1$ (taux d'accroissement naturel des lapins), $\beta = 0.1$ (taux de mortalité des lapins par prédation), $\delta = 1.5$ (taux de mortalité des renards sans source de nourriture), $\gamma = 0.75$ (combien de lapins sont nécessaires pour que les renards se reproduisent).
Calculez les valeurs propres de la matrice Jacobienne des points pour déterminer leur équilibre :
$\begin{bmatrix}0\\0\end{bmatrix}$ et $\begin{bmatrix}\gamma/(\delta * \beta)\\ \alpha/\beta\end{bmatrix}$
Intégrez le problème et représentez graphiquement les populations.

6.15 Trop de bruit

Concepts : 1) traitement du signal en 1D 2) `FFT`

Étapes :

1. générez un signal comme une somme de cos de différentes fréquences
2. effectuez la FFT de ce signal
3. coupez les basses fréquences de ce signal
4. affichez le signal débruité

6.16 Simplifions

Concepts : 1) sympy 2) calcul symbolique

Calculez le coefficient de $x^3y^7z^2$ dans $(x + y+)^{12}$

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