

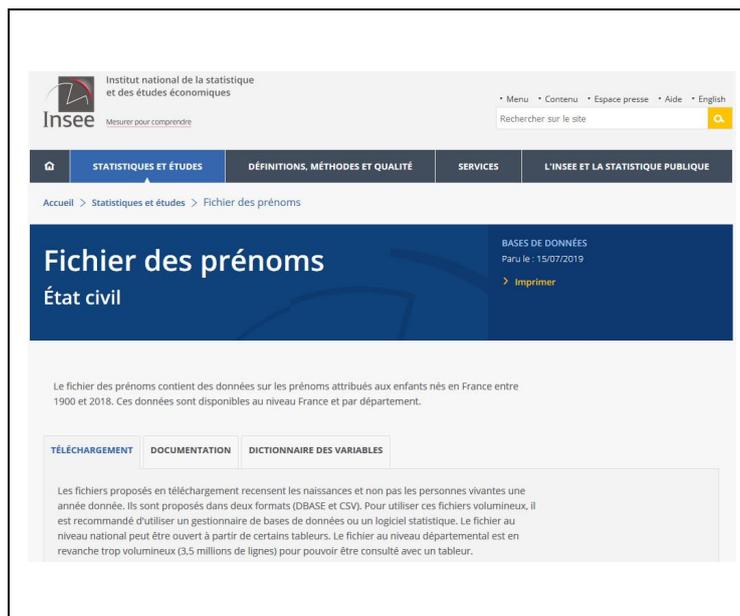
Activité Python : Prénoms

Objectif : Découvrir la lecture/écriture dans un fichier en Python.

Vous êtes chargés par des jeunes couples de les aider à trouver un prénom pour leur futur bébé ! Ils souhaitent tester l'originalité de leurs idées initiales de prénoms, garçons et filles.

Vous repérez l'existence du fichier national des naissances en France, disponible à l'adresse suivante :



	<h2>Fichier des prénoms : État civil</h2> <p>https://www.insee.fr/fr/statistiques/2540004</p> <p>Les fichiers proposés en téléchargement recensent les naissances et non pas les personnes vivantes une année donnée. Ils sont proposés dans deux formats (DBASE et CSV). Pour utiliser ces fichiers volumineux, il est recommandé d'utiliser un gestionnaire de bases de données ou un logiciel statistique. Le fichier au niveau national peut être ouvert à partir de certains tableurs. Le fichier au niveau départemental est en revanche trop volumineux (3,5 millions de lignes) pour pouvoir être consulté avec un tableur.</p>
--	--

Info en détail (Extrait de la documentation)

1. Le premier fichier national comporte 636 474 enregistrements et quatre variables décrites ci-après.

Ce fichier est trié selon les variables SEXE, PREUSUEL, ANNAIS.

- SEXE - intitulé : sexe - Type : caractère - Longueur : 1 - Modalité : 1 pour masculin, 2 pour féminin
- PREUSUEL - intitulé : premier prénom - Type : caractère - Longueur : 25
- ANNAIS - intitulé : année de naissance - Type : caractère - Longueur : 4 - Modalité : 1900 à 2018, XXXX
- NOMBRE - intitulé : fréquence - Type : numérique - Longueur : 8

2. Le second fichier départemental comporte 3 624 994 enregistrements et cinq variables décrites ci-après.

Ce fichier est trié selon les variables SEXE, PREUSUEL, ANNAIS, DPT.

- SEXE - intitulé : sexe - Type : caractère - Longueur : 1 - Modalité : 1 pour masculin, 2 pour féminin
- PREUSUEL - intitulé : premier prénom - Type : caractère - Longueur : 25
- ANNAIS - intitulé : année de naissance - Type : caractère - Longueur : 4 - Modalité : 1900 à 2018, XXXX
- DPT - intitulé : département de naissance - Type : caractère - Longueur : 3 - Modalité : liste des départements, XX
- NOMBRE - intitulé : fréquence - Type : numérique - Longueur : 8

Nous allons donner quelques éléments de cours avant d'aider ces parents avec les prénoms !

Intérêt de lire et écrire dans des fichiers :

En fermant un programme python, aucune variable n'est conservée. Les fichiers sont un bon moyen de garder les valeurs de certaines variables (ou objets) pour pouvoir les récupérer en rouvrant le programme. Par exemple, pour conserver les statistiques d'un joueur, ou bien le score maximal obtenu, etc, pour ne pas rejouer en repartant de zéro.

Comment lire dans un fichier ?

A vous de jouer.

Créez un fichier « fichier.txt » à coté du script python, à l'aide du bloc-note de windows ou d'un autre éditeur de texte. Ecrivez du texte dedans, enregistrez et fermez le fichier.



« Je découvre la lecture et l'écriture dans les fichiers ! »

Et maintenant, nous allons lire ce fichier depuis un programme Python.

On utilise la fonction `open(chemin, mode)`, disponible sans avoir besoin de rien importer. Elle prend en paramètre :

- Le chemin menant au fichier à ouvrir (pas de chemin si le fichier est dans le même dossier que le script Python);
- Le mode d'ouverture.

Le mode est donné sous la forme d'une chaîne de caractères. Voici les principaux modes :

- `'r'`: ouverture en lecture (r comme « read »).
- `'w'`: ouverture en écriture (w comme « write »). Le contenu du fichier est écrasé. Si le fichier n'existe pas, il est créé.
- `'a'`: ouverture en écriture en mode ajout (a comme « append », qui signifie « ajout à la suite »). On écrit à la fin du fichier sans écraser l'ancien contenu du fichier. Si le fichier n'existe pas, il est créé.

```
mon_fichier = open("fichier.txt", "r")
```

On va maintenant lire l'intégralité du fichier. Pour ce faire, on utilise la méthode `read`, qui renvoie l'intégralité du fichier :

```
contenu = mon_fichier.read()  
print(contenu)
```

La méthode `read` renvoie tout le contenu du fichier, que l'on capture dans une chaîne de caractères. Notre fichier ne contient pas de saut de ligne mais, si c'était le cas, vous auriez dans votre variable `contenu` les signes `\n` traduisant un saut de ligne.

Si l'on souhaite lire une ligne seulement, on utilise la méthode `readline()`. En l'utilisant à nouveau, cela permet de passer à la ligne suivante.

Il faut aussi bien penser à fermer le fichier après l'avoir ouvert. Si d'autres applications, ou d'autres morceaux de votre propre code, souhaitent accéder à ce fichier, ils ne pourront pas car le fichier sera déjà ouvert. C'est surtout vrai en écriture, mais prenez de bonnes habitudes. La méthode à utiliser est `close`.

```
mon_fichier.close()
```

On teste le tout ?

```
tuto_eleve.py x
1 mon_fichier = open("fichier.txt", "r")
2 contenu = mon_fichier.read()
3 print(contenu)
4 mon_fichier.close()

Shell x
>>> %Run tuto_eleve.py
Je découvre la lecture et l'écriture dans les fichiers !
>>> |
```

```
mon_fichier = open("fichier.txt",
"r")
contenu = mon_fichier.read()
print(contenu)
mon_fichier.close()
```

Et comment écrire dans un fichier ?

Bien entendu, il nous faut ouvrir le fichier avant tout. Vous pouvez utiliser le mode `w` ou le mode `a`. Le premier écrase le contenu éventuel du fichier, alors que le second ajoute ce que l'on écrit à la fin du fichier :

- Si je veux stocker le score maximal atteint par un joueur,, on va supprimer l'ancienne valeur en utilisant le mode `w`.
- Si l'on veut enregistrer la liste des personnes qui ont réussi à finir le jeu, on peut simplement vouloir ajouter un nouveau nom à la suite des autres : on prend le mode `a`.

À vous de voir en fonction de vos besoins. Dans tous les cas, ces deux modes créent le fichier s'il n'existe pas.

Pour écrire dans un fichier, on utilise la méthode `write` en lui passant en paramètre la chaîne à écrire dans le fichier. Elle renvoie le nombre de caractères qui ont été écrits. On n'est naturellement pas obligé de récupérer cette valeur, sauf si on en a besoin.

```
mon_fichier = open("fichier.txt", "w") #On ouvre et on efface tout ! (mode w)
mon_fichier.write("Test d'écriture dans un fichier via Python")
mon_fichier.close()
```

Vous pouvez vérifier que votre fichier contient bien le texte qu'on y a écrit :

```
tuto_eleve.py x
1 mon_fichier = open("fichier.txt", "w") #On ouvre et on efface tout ! (mode w)
2 mon_fichier.write("Test d'écriture dans un fichier via Python")
3 mon_fichier.close()
4

fichier.txt - Bloc-notes
Fichier Edition Format Affichage ?
Test d'écriture dans un fichier via Python

Shell x
>>> %Run tuto_eleve.py
Je découvre la lecture et l'écriture dans les fichiers !
>>> %Run tuto_eleve.py
>>>
```

Informations complémentaires :

- La méthode `write` n'accepte en paramètre que des chaînes de caractères. Si vous voulez écrire dans votre fichier des nombres, comme le score maximum donné en exemple, il vous faudra les convertir en chaîne avant de les écrire et les convertir en entier après les avoir lus.
- Si vous avez besoin de davantage de fonctions pour manipuler des fichiers et des dossiers, allez voir du côté du module `os` !

Concluons :

1. On peut ouvrir un fichier en utilisant la fonction `open` prenant en paramètre le chemin vers le fichier et le mode d'ouverture.
2. On peut lire dans un fichier en utilisant la méthode `read`.
3. On peut écrire dans un fichier en utilisant la méthode `write`.
4. Un fichier doit être refermé après usage en utilisant la méthode `close`.

tuto_eleve.py ×

```
1 mon_fichier = open("fichier.txt", "r")
2 contenu = mon_fichier.read()
3 print(contenu)
4 mon_fichier.close()
```

Shell ×

```
>>> %Run tuto_eleve.py
```

```
Je découvre la lecture et l'écriture dans les fichiers !
```

```
>>> |
```

Un code un peu plus complet à analyser !

On va répondre au besoin exprimé au début de notre document : les parents cherchent des statistiques pour les aider dans le choix de prénoms pour leurs enfants. Récupérez le fichier `nat2018.csv` mentionné à la première page et tentez de calculer combien d'enfants on eu votre prénom entre 1900 et 2018 !

Voici un exemple de code fonctionnel, avec beaucoup de commentaires pour vous aider à comprendre.

```
# -*-coding:utf-8 -*

prenom = str(input("Bonjour,\nEntrez un prénom svp : ")).upper() #le prénom
voulu, en majuscules
genre = int(input("Entrez 1 pour un garçon, 2 pour une fille : "))
somme_prenoms = 0 #On initialise une variable pour compter le total de naissance
f = open("nat2018.csv", encoding="utf-8", mode="r") # Ouverture du fichier source
en lecture
enregistreur = open(prenom+".txt", "w") # Création du fichier de sauvegarde
enregistreur.write("Détail des naissances en France entre 1900 et 2018 des
enfants nommés "+prenom+".\n") #On écrit une intro en début de fichier
ligne = f.readline()#On lit la première ligne

while ligne != "":
    sexe , preusuel , annais , nombre = ligne.split(";")#On sépare les données de
la ligne en analysant ce qui se trouve entre les points virgules
    if (preusuel == prenom and int(sexe) == genre): #Si la ligne nous concerne
        somme_prenoms = somme_prenoms + int(nombre) #On ajoute les enfant de cette
année au compteur
        enregistreur.write("En "+annais+": "+str(int(nombre))+ " enfants.\n")
        ligne = f.readline()#On passe à la ligne suivante
    enregistreur.write("\nNombre de "+prenom+" né(e)s en France entre 1900 et 2018 :
" + str(somme_prenoms)+" enfants.\n")

enregistreur.write("Source du fichier des prénoms :
https://www.insee.fr/fr/statistiques/2540004\n")
enregistreur.write("Copyright Marie Pila SNT 2019-2020 ;)\n")
print("Nombre de", prenom, "né(e)s en France entre 1900 et 2018 : ",
somme_prenoms, " enfants.")
print("Source du fichier des prénoms :
https://www.insee.fr/fr/statistiques/2540004")
print("Fin du programme")
f.close()
enregistreur.close()
```



```
ALICE.txt - Bloc-notes
Fichier Edition Format Affichage ?
Détail des naissances en France entre 1900 et 2018 des enfants
nommés ALICE.
En 1900: 2939 enfants.
En 1901: 3150 enfants.
En 1902: 3188 enfants.
En 1903: 3283 enfants.
En 1904: 3357 enfants.
En 1905: 3516 enfants.
En 1906: 3639 enfants.
En 1907: 3512 enfants.
En 1908: 3745 enfants.
En 1909: 3700 enfants.
En 1910: 3599 enfants.
En 1911: 3385 enfants.
```

Bonus pour les plus braves : tracez des graphiques avec tout cela !

```
# coding: utf-8

import matplotlib.pyplot

def ajout_courbe(prenom,genre):
    annees, naissances = [],[]
    f = open("nat2018.csv", encoding="utf-8", mode="r") # Ouverture du fichier
    source en lecture
    ligne = f.readline()
    while ligne != "": #On parcourt le fichier tant qu'on n'est pas arrivé à la
fin du fichier (ligne vide)
        sexe , preusuel , annais , nombre = ligne.split(";")
        if (preusuel == prenom and int(sexe) == genre and annais != "XXXX"):
            #C'est le bon prénom, genre, et on connaît l'année : on ajoute dans
le graphique
            annees.append(annais)
            naissances.append(int(nombre))
        ligne = f.readline() #On avance d'une ligne dans le fichier
    f.close()
    matplotlib.pyplot.plot(annees,naissances, label=prenom) #On ajout la courbe,
avec les années en "x" et les naissances en "y"

#Partie principale du programme
print("Bonjour,")
annees, naissances = [],[]
while True: #Boucle infinie
    prenom = str(input("Ajoutez un prénom svp (ou laissez vide pour finir):
")).upper() #le prénom en majuscules
    if prenom == "":
        break #Si aucun prénom n'est donné, on stoppe la boucle
    genre = int(input("Entrez 1 pour un garçon, 2 pour une fille : ") or "1")#Si
rien n'est rentré, on met une valeur 1 par défaut
    ajout_courbe(prenom,genre) #On appelle la fonction en envoyant les 2
paramètres

matplotlib.pyplot.title("Nombre d' enfants par année")
matplotlib.pyplot.xlabel('Année de naissance')
matplotlib.pyplot.ylabel('Nombre d\'enfants')
matplotlib.pyplot.legend() #Ajout de la légende
#matplotlib.pyplot.grid(True) #Quadrillage éventuellement
matplotlib.pyplot.show() #Affichage du graphique final
print("Fin du programme")
```

