

## 8.2

### Les fichiers csv

NSI 1ÈRE - JB DUTHOIT

#### 8.2.1 Les fichiers .csv

L'organisation des données en tableau est très courante et très ancienne.

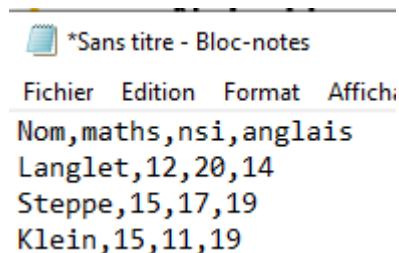
► Le bulletin d'un élève, les listes de classe par exemple sont organisés en table. Un fichier **csv** (pour comma separated values, soit en français valeurs séparées par des virgules) est un format très pratique pour représenter des données structurées.

1. C'est un fichier texte

- Chaque ligne du texte correspond à une ligne du tableau et les virgules correspondent aux séparations entre les colonnes.
- Toutes les lignes du fichier ont le même nombre de champs (colonnes).

#### Exercice 8.150

Créer un fichier `NotesEleves.txt` avec le bloc-notes du type :



```
*Sans titre - Bloc-notes
Fichier Edition Format Affichage
Nom,maths,nsi,anglais
Langlet,12,20,14
Steppe,15,17,19
Klein,15,11,19
```

#### Exercice 8.151

Visiter le site <https://www.data.gouv.fr/>, puis trouver le fichier qui donne la liste des pays du monde et leur capitale respective. Télécharger-le sur votre ordinateur. Ouvrir votre fichier avec le bloc-notes.



```
Fichier Edition Format Affichage Aide
NOM;NOM_ALPHA;CODE;ARTICLE;NOM_LONG;CAPITALE
Afghanistan;Afghanistan;AFG;l';République islamique d'Afghanistan;Kaboul
Afrique du Sud;Afrique du Sud;ZAF;l';République d'Afrique du Sud;Prétoria
Albanie;Albanie;ALB;l';République d'Albanie;Tirana
Algérie;Algérie;DZA;l';République algérienne démocratique et populaire;Alger
Allemagne;Allemagne;DEU;l';République fédérale d'Allemagne;Berlin
Andorre;Andorre;AND;l';Principauté d'Andorre;Andorre-la-Vieille
Angola;Angola;AGO;l';République d'Angola;Luanda
Antigua-et-Barbuda;Antigua-et-Barbuda;ATG;;Antigua-et-Barbuda;Saint John's
Arabie saoudite;Arabie saoudite;SAU;l';Royaume d'Arabie saoudite;Riyad
Argentine:Argentine:ARG:l':République argentine:Buenos Aires
```

► Ici, c'est le "point virgule" qui a été utilisé comme séparateur.

**⚠️ Attention !**

La virgule est un standard pour les données anglo-saxonnes, mais pas pour les données aux normes françaises.

En effet, en français, la virgule est le séparateur des chiffres décimaux. Il serait impossible de différencier les virgules des décimaux et les virgules de séparation des informations. C'est pourquoi on utilise un autre séparateur : le point-virgule ( ; ).

➡️ Dans certains cas cela peut engendrer quelques problèmes, vous devrez donc rester vigilants sur le type de séparateur utilisé.

"NOM", "NOM\_ALPHA", "CODE", "ARTICLE"… sont appelés des **descripteurs** alors que, par exemple, "Afghanistant", "Afghanistan" et "AFG", "1'" sont les **valeurs du descripteur** "NOM".

### 8.2.2 Importer des données depuis un fichier csv avec Python

- Méthode 1 :

```
import csv
pays = []
with open("pays.csv", "r") as fichier:
    table = csv.reader(fichier, delimiter = ';')
    for ligne in table:
        pays.append(ligne)
```

On a ainsi une liste de listes

- Quel est le nombre de pays présents dans cette liste ?
- Afficher le premier élément de la liste pays.
- Et afficher le 110 ème

```
>>> pays
[[['countryCode', 'countryName', 'area', 'population', 'continent', 'currencyCode', 'curren
'capital'], ['AD', 'Andorra', '468', '84000', 'EU', 'EUR', 'Euro', 'Andorra la Vella'], [
ted Arab Emirates', '82880', '4975593', 'AS', 'AED', 'Dirham', 'Abu Dhabi'], ['AF', 'Afgha
'647500', '29121286', 'AS', 'AFN', 'Afghani', 'Kabul'], ['AG', 'Antigua and Barbuda', '441
'1000000', '2700000', 'NA', 'USD', 'US Dollar', 'St. John's'], ['AI', 'Anguilla', '240', '76000
'NA', 'USD', 'US Dollar', 'The Valley'], ['AQ', 'AQ', '240', '1000000', 'NA', 'USD', 'US D
'1000000', '1000000', 'NA', 'USD', 'US Dollar', 'Dili'], ['AS', 'American Samoa', '240', '100000
'NA', 'USD', 'US Dollar', 'Pago Pago'], ['AT', 'Austria', '80459', '8400000', 'EU', 'EUR', 'E
'1000000', '1000000', 'NA', 'USD', 'US Dollar', 'Nuku'alofa'], ['AU', 'Australia', '7692024
'1000000', '1000000', 'NA', 'USD', 'US Dollar', 'Canberra'], ['AZ', 'Azerbaijan', '85940', '9000000
'NA', 'USD', 'US Dollar', 'Bakı'], ['BA', 'Bosnia and Herzegovina', '51129', '3800000', 'EU
'1000000', '1000000', 'NA', 'USD', 'US Dollar', 'Sarajevo'], ['BH', 'Bahrain', '26521', '700000
'NA', 'USD', 'US Dollar', 'Manama'], ['BD', 'Bangladesh', '1484000', '150000000', 'EU', 'BDT', 'B
'1000000', '1000000', 'NA', 'USD', 'US Dollar', 'Dhaka'], ['BF', 'Burkina Faso', '2742700', '16000000
'NA', 'USD', 'US Dollar', 'Ouagadougou'], ['BG', 'Bulgaria', '110994', '7000000', 'EU', 'BGN', 'B
'1000000', '1000000', 'NA', 'USD', 'US Dollar', 'Sofia'], ['BH', 'Bahrain', '26521', '700000
'NA', 'USD', 'US Dollar', 'Manama'], ['BI', 'Burundi', '278400', '10000000', 'NA', 'BIF', 'B
'1000000', '1000000', 'NA', 'USD', 'US Dollar', 'Bujumbura'], ['BT', 'Bhutan', '38394', '700000
'NA', 'USD', 'US Dollar', 'Thimphu'], ['BW', 'Botswana', '2782000', '20000000', 'SA', 'BWP', 'B
'1000000', '1000000', 'NA', 'USD', 'US Dollar', 'Gaborone'], ['BV', 'Bouvet Island', '0', '0', 'NA
'NA', 'NA', 'NA'], ['BY', 'Belarus', '2000000', '9000000', 'EU', 'BYN', 'Belarusian Ruble', 'Minsk
'1000000', '1000000', 'NA', 'USD', 'US Dollar', 'Minsk'], ['BZ', 'Belize', '22700', '3000000
'NA', 'USD', 'US Dollar', 'Belmopan'], ['CL', 'Chile', '356300', '17000000', 'SA', 'CLP', 'Chilean Peso
'1000000', '1000000', 'NA', 'USD', 'US Dollar', 'Santiago'], ['CO', 'Colombia', '4447800', '45000000
'NA', 'USD', 'US Dollar', 'Bogotá'], ['CR', 'Costa Rica', '51020', '4500000', 'NA', 'CRC', 'Costa Rican Colón', 'San José
'1000000', '1000000', 'NA', 'USD', 'US Dollar', 'San José'], ['CY', 'Cyprus', '92450', '10000000
'NA', 'USD', 'US Dollar', 'Nicosia'], ['CZ', 'Czech Republic', '42847', '10000000', 'EU', 'CZK', 'Czech Koruna', 'Prague
'1000000', '1000000', 'NA', 'USD', 'US Dollar', 'Prague'], ['DK', 'Democratic Republic of Congo
'1000000', '1000000', 'NA', 'USD', 'US Dollar', 'Kinshasa'], ['DO', 'Dominican Republic', '48440
'1000000', '1000000', 'NA', 'USD', 'US Dollar', 'Santo Domingo'], ['DZ', 'Algeria', '2381700', '35000000
'NA', 'DZD', 'Algerian Dinar', 'Algiers'], ['EC', 'Ecuador', '252400', '15000000', 'SA', 'ECU', 'Ecuadorian Sucre', 'Quito
'1000000', '1000000', 'NA', 'USD', 'US Dollar', 'Quito'], ['EE', 'Estonia', '45226', '10000000
'NA', 'USD', 'US Dollar', 'Tallinn'], ['EG', 'Egypt', '1000000', '80000000', 'NA', 'EGP', 'Egyptian Pound', 'Cairo
'1000000', '1000000', 'NA', 'USD', 'US Dollar', 'Cairo'], ['ES', 'Spain', '499900', '45000000
'NA', 'USD', 'US Dollar', 'Madrid'], ['FI', 'Finland', '134100', '5000000', 'EU', 'EUR', 'Euro', 'Helsinki
'1000000', '1000000', 'NA', 'USD', 'US Dollar', 'Helsinki'], ['FO', 'Faroe Islands', '1390', '50000
'NA', 'USD', 'US Dollar', 'Tórshavn'], ['GR', 'Greece', '131992', '10000000', 'EU', 'EUR', 'Euro', 'Athens
'1000000', '1000000', 'NA', 'USD', 'US Dollar', 'Athens'], ['HR', 'Croatia', '56620', '4000000
'NA', 'USD', 'US Dollar', 'Zagreb'], ['HU', 'Hungary', '93030', '9000000', 'EU', 'HUF', 'Hungarian Forint', 'Budapest
'1000000', '1000000', 'NA', 'USD', 'US Dollar', 'Budapest'], ['ID', 'Indonesia', '1904500', '250000000
'NA', 'USD', 'US Dollar', 'Jakarta'], ['IE', 'Ireland', '70270', '4000000', 'EU', 'EUR', 'Euro', 'Dublin
'1000000', '1000000', 'NA', 'USD', 'US Dollar', 'Dublin'], ['IL', 'Israel', '20500', '7000000
'NA', 'USD', 'US Dollar', 'Tel Aviv'], ['IQ', 'Iraq', '460000', '30000000', 'NA', 'IQD', 'Iraqi Dinar', 'Baghdad
'1000000', '1000000', 'NA', 'USD', 'US Dollar', 'Baghdad'], ['IS', 'Iceland', '33996', '10000000
'NA', 'USD', 'US Dollar', 'Reykjavík'], ['IT', 'Italy', '301330', '60000000', 'EU', 'EUR', 'Euro', 'Rome
'1000000', '1000000', 'NA', 'USD', 'US Dollar', 'Rome'], ['JO', 'Jordan', '89240', '6000000
'NA', 'USD', 'US Dollar', 'Amman'], ['KZ', 'Kazakhstan', '2724900', '180000000', 'EU', 'KZT', 'Kazakh Tenge', 'Astana
'1000000', '1000000', 'NA', 'USD', 'US Dollar', 'Astana'], ['LB', 'Lebanon', '102200', '40000000
'NA', 'USD', 'US Dollar', 'Beirut'], ['LT', 'Lithuania', '25210', '10000000', 'EU', 'EUR', 'Euro', 'Vilnius
'1000000', '1000000', 'NA', 'USD', 'US Dollar', 'Vilnius'], ['LV', 'Latvia', '20494', '10000000
'NA', 'USD', 'US Dollar', 'Riga'], ['MD', 'Moldova', '33840', '10000000', 'EU', 'MDL', 'Moldovan Leu', 'Chișinău
'1000000', '1000000', 'NA', 'USD', 'US Dollar', 'Chișinău'], ['ME', 'Montenegro', '13810', '10000000
'NA', 'USD', 'US Dollar', 'Podgorica'], ['MT', 'Malta', '40270', '10000000', 'EU', 'EUR', 'Euro', 'Valletta
'1000000', '1000000', 'NA', 'USD', 'US Dollar', 'Valletta'], ['NL', 'Netherlands', '166300', '150000000
'NA', 'USD', 'US Dollar', 'The Hague'], ['NO', 'Norway', '24220', '40000000', 'EU', 'NOK', 'Norwegian Krone', 'Oslo
'1000000', '1000000', 'NA', 'USD', 'US Dollar', 'Oslo'], ['PK', 'Pakistan', '1942000', '180000000
'NA', 'PKR', 'Pakistani Rupee', 'Islamabad'], ['PS', 'Palestinian Territories', '460000', '10000000
'NA', 'USD', 'US Dollar', 'Ramallah'], ['PT', 'Portugal', '1000000', '10000000', 'EU', 'EUR', 'Euro', 'Lisbon
'1000000', '1000000', 'NA', 'USD', 'US Dollar', 'Lisbon'], ['RO', 'Romania', '233300', '18000000
'NA', 'USD', 'US Dollar', 'Bucharest'], ['RS', 'Serbia', '7000000', '10000000', 'EU', 'RSD', 'Serbian Dinar', 'Belgrade
'1000000', '1000000', 'NA', 'USD', 'US Dollar', 'Belgrade'], ['SI', 'Slovenia', '20250', '10000000
'NA', 'USD', 'US Dollar', 'Ljubljana'], ['SK', 'Slovakia', '54650', '10000000', 'EU', 'EUR', 'Euro', 'Bratislava
'1000000', '1000000', 'NA', 'USD', 'US Dollar', 'Bratislava'], ['TR', 'Turkey', '75000000', '700000000
'NA', 'USD', 'US Dollar', 'Ankara'], ['UA', 'Ukraine', '27000000', '450000000', 'EU', 'UAH', 'Ukrainian Hryvnia', 'Kyiv
'1000000', '1000000', 'NA', 'USD', 'US Dollar', 'Kyiv'], ['YE', 'Yemen', '2140000', '10000000
'NA', 'USD', 'US Dollar', 'Sana'a']]
```

Cette solution présente des inconvénients.

- La 1ère ligne, celle des attributs, a été chargée comme une ligne de données.
- Toutes les données ont été converties en chaînes de caractères même les entiers.
- Plus gênant, le lien entre les valeurs du tableau pays[1] et le nom des enregistrements, contenus dans pays[0], n'est pas direct.
- Nous allons donc utiliser une seconde méthode !

- Méthode 2 :

```

1 import csv
2 pays = []
3 with open("pays.csv","r") as fichier:
4     table = csv.DictReader(fichier, delimiter = ';')
5     for ligne in table:
6         pays.append(ligne)

```

```

>>> pays
[OrderedDict([('countryCode', 'AD'), ('countryName', 'Andorra'), ('area', '468'), ('population', '84000'), ('continent', 'EU'), ('currencyCode', 'EUR'), ('currencyName', 'Euro'), ('ital', 'Andorra la Vella')]), OrderedDict([('countryCode', 'AE'), ('countryName', 'United ab Emirates'), ('area', '82880'), ('population', '4975593'), ('continent', 'AS'), ('curre Code', 'AED'), ('currencyName', 'Dirham'), ('capital', 'Abu Dhabi')]), OrderedDict([('couyCode', 'AF'), ('countryName', 'Afghanistan'), ('area', '647500'), ('population', '291212'), ('continent', 'AS'), ('currencyCode', 'AFN'), ('currencyName', 'Afghani'), ('capital', 'abul')]), OrderedDict([('countryCode', 'AG'), ('countryName', 'Antigua and Barbuda'), ('a

```

Examinons les avantages de cette solution :

- La 1ère ligne, celle des attributs, n'a pas été chargée comme une ligne de données.
- Le lien entre les valeurs du tableau pays[1] et le nom des enregistrements, contenus dans pays[0], est direct.
- Quel est la longueur de la liste ? Est-ce en accord avec le nombre de pays ? Expliquer
- .

Mais :

- toutes les données restent converties en chaînes de caractères même les entiers.
- Le format d'affichage peut déconcerter

☛ Nous allons donc utiliser une autre amélioration !

- Méthode 3 :

```

import csv
pays = []
with open("pays.csv","r") as fichier:
    table = csv.DictReader(fichier, delimiter = ';')
    for ligne in table:
        pays.append(dict(ligne))

```

```
>>> pays
[{'countryCode': 'AD', 'countryName': 'Andorra', 'area': '468', 'population': '84000', 'continent': 'EU', 'currencyCode': 'EUR', 'currencyName': 'Euro', 'capital': 'Andorra la Vella'}, {'countryCode': 'AE', 'countryName': 'United Arab Emirates', 'area': '82880', 'population': '4975593', 'continent': 'AS', 'currencyCode': 'AED', 'currencyName': 'Dirham', 'capital': 'Abu Dhabi'}, {"countryCode": "AF", "countryName": "Afghanistan", "area": "647500", "population": "29121286", "continent": "AS", "currencyCode": "AFN", "currencyName": "Afghani", "capital": "Kabul"}, {"countryCode": "AG", "countryName": "Antigua and Barbuda", "area": "443", "population": "86754", "continent": "NA", "currencyCode": "XCD", "currencyName": "Dollar", "capital": "St. John's"}, {"countryCode": "AI", "countryName": "Anguilla", "area": "102", "population": "102", "continent": "NA", "currencyCode": "XCD", "currencyName": "Dollar", "capital": "The Valley"}]
```

Une liste de dico, c'est beaucoup plus lisible!

### Exercice 8.152

Ajouter quelques lignes au code précédent afin de convertir les str en entiers ou en flottants