C7 Lecture et traitement de données en tables

Données en tables

- Le traitement et l'analyse de données volumineuses (*big data*) est l'une des activités principales en informatique de nos jours .

C7 Lecture et traitement de données en tables

Données en tables

- Le traitement et l'analyse de données volumineuses (*big data*) est l'une des activités principales en informatique de nos jours .
- Ces données sont souvent organisées en tables.

C7 Lecture et traitement de données en tables

Données en tables

- Le traitement et l'analyse de données volumineuses (*big data*) est l'une des activités principales en informatique de nos jours .
- Ces données sont souvent organisées en tables.
- Une ligne de données en table s'appelle un enregistrement

C7 Lecture et traitement de données en tables

Données en tables

- Le traitement et l'analyse de données volumineuses (*big data*) est l'une des activités principales en informatique de nos jours .
- Ces données sont souvent organisées en tables.
- Une ligne de données en table s'appelle un enregistrement
- Une colonne s'apelle un champ

C7 Lecture et traitement de données en tables

Données en tables

- Le traitement et l'analyse de données volumineuses (*big data*) est l'une des activités principales en informatique de nos jours .
- Ces données sont souvent organisées en tables.
- Une ligne de données en table s'appelle un enregistrement
- Une colonne s'apelle un champ
- Les titres des colonnes sont les descripteurs

C7 Lecture et traitement de données en tables

Données en tables

- Le traitement et l'analyse de données volumineuses (*big data*) est l'une des activités principales en informatique de nos jours .
- Ces données sont souvent organisées en tables.
- Une ligne de données en table s'appelle un enregistrement
- Une colonne s'apelle un champ
- Les titres des colonnes sont les descripteurs

Format csv

Le format de fichier csv (*comma separated value*) représente des données en tables. Chaque ligne du fichier est une donnée et sur chaque ligne les champs sont séparées par des virgules (ou parfois un autre caractère comme le point-virgule).

C7 Lecture et traitement de données en tables

Exemple

- Des données en table

Nom	Prénom	Naissance
Pascal	Blaise	1623
Lovelace	Ada	1815
Boole	George	1815

- Représentation en fichier csv

```
Nom;Prénom;Naissance
Pascal;Blaise;1623
Lovelace;Ada;1815
Boole;George;1815
```

Le fichier csv à droite sera utilisé par la suite, on l'appelle `exemple.csv` de façon à y faire référence.

C7 Lecture et traitement de données en tables

Exemple

- Des données en table

Nom	Prénom	Naissance
Pascal	Blaise	1623
Lovelace	Ada	1815
Boole	George	1815

- Représentation en fichier csv

```
Nom;Prénom;Naissance
Pascal;Blaise;1623
Lovelace;Ada;1815
Boole;George;1815
```

Le fichier csv à droite sera utilisé par la suite, on l'appelle `exemple.csv` de façon à y faire référence.

Remarques

- La première ligne du fichier csv décrit les champs, il contient les attributs (appelés aussi descripteur).

C7 Lecture et traitement de données en tables

Exemple

- Des données en table

Nom	Prénom	Naissance
Pascal	Blaise	1623
Lovelace	Ada	1815
Boole	George	1815

- Représentation en fichier csv

```
Nom;Prénom;Naissance
Pascal;Blaise;1623
Lovelace;Ada;1815
Boole;George;1815
```

Le fichier csv à droite sera utilisé par la suite, on l'appelle `exemple.csv` de façon à y faire référence.

Remarques

- La première ligne du fichier csv décrit les champs, il contient les attributs (appelés aussi descripteur).
- Les données d'un fichier csv sont au format texte, par conséquent même une donnée numérique (comme ici l'année de naissance) est en fait une chaine de caractères.

C7 Lecture et traitement de données en tables

Python et les fichiers csv

- Le module `csv` de Python permet de récupérer les informations d'un fichier csv, sous forme de listes de listes ou de dictionnaires

C7 Lecture et traitement de données en tables

Python et les fichiers csv

- Le module `csv` de Python permet de récupérer les informations d'un fichier csv, sous forme de listes de listes ou de dictionnaires
- Pour les dictionnaires, ce sont alors les descripteurs qui servent de clés.

C7 Lecture et traitement de données en tables

Python et les fichiers csv

- Le module `csv` de Python permet de récupérer les informations d'un fichier csv, sous forme de listes de listes ou de dictionnaires
- Pour les dictionnaires, ce sont alors les descripteurs qui servent de clés.
- Tous les champs (même ceux contenant des nombres) sont récupérés sous forme de chaines de caractères (type `str` de Python) à la façon de ce qui se passe lors d'un `input`. Faire donc attention lors de calculs ou de comparaisons avec les données de ces champs.

C7 Lecture et traitement de données en tables

Exemple

Récupération des éléments du fichier `exemple.csv` ci-dessus dans un dictionnaire :

```
1 import csv
2 fic=open("exemple.csv","r",encoding="utf-8")
3 # Lecture sous forme de dictionnaire
4 donnees = list(csv.DictReader(fic,delimiter=';'))
5 fic.close()
```

C7 Lecture et traitement de données en tables

Exemple

Récupération des éléments du fichier `exemple.csv` ci-dessus dans un dictionnaire :

```
1 import csv
2 fic=open("exemple.csv","r",encoding="utf-8")
3 # Lecture sous forme de dictionnaire
4 donnees = list(csv.DictReader(fic,delimiter=';'))
5 fic.close()
```

Après execution, on a par exemple

`donnees[0]={'Nom' : 'Pascal', 'Prenom' : 'Blaise', 'Naissance' : '1623'}`

C7 Lecture et traitement de données en tables

Exemple

Récupération des éléments du fichier `exemple.csv` ci-dessus dans un dictionnaire :

```
1 import csv
2 fic=open("exemple.csv","r",encoding="utf-8")
3 # Lecture sous forme de dictionnaire
4 donnees = list(csv.DictReader(fic,delimiter=';'))
5 fic.close()
```

Après execution, on a par exemple
`donnees[0]={'Nom' : 'Pascal', 'Prenom' : 'Blaise', 'Naissance' : '1623'}`
C'est à dire que chaque ligne de la table correspond à un dictionnaire

C7 Lecture et traitement de données en tables

Traitement des données

- Une fois les données `csv` lues et récupérées dans un dictionnaire, on peut trier les informations et y faire des recherches.

C7 Lecture et traitement de données en tables

Traitement des données

- Une fois les données csv lues et récupérées dans un dictionnaire, on peut trier les informations et y faire des recherches.
- Par exemple pour le fichier csv donné en exemple :

```
Nom;Prénom;Naissance
Pascal;Blaise;1623
Lovelace;Ada;1815
Boole;George;1815
```

C7 Lecture et traitement de données en tables

Traitement des données

- Une fois les données csv lues et récupérées dans un dictionnaire, on peut trier les informations et y faire des recherches.
- Par exemple pour le fichier csv donné en exemple :

```
Nom;Prénom;Naissance
Pascal;Blaise;1623
Lovelace;Ada;1815
Boole;George;1815
```

- Si les données sont récupérées dans la liste de dictionnaire `personnages`. On peut afficher les personnes nées en 1815 avec :

```
1       for p in personnages:
2           if p["Naissance"]=="1815":
3               print(p["Nom],p["Prénom"])
```

Trier une liste en Python

- La fonction `sorted` de Python permet de trier une liste. Elle renvoie la liste triée. La syntaxe est la suivante : `liste_triee = sorted(liste)`.

C7 Lecture et traitement de données en tables

Trier une liste en Python

- La fonction `sorted` de Python permet de trier une liste. Elle renvoie la liste triée. La syntaxe est la suivante : `liste_triee = sorted(liste)`.
- Par exemple :

```
1 notes = [15,11,10,18,9]
2 note_triees=sort(notes)
3 print(notes_triees)
```

affichera : `[9,10,11,15,18]`

Trier une liste en Python

- La fonction `sorted` de Python permet de trier une liste. Elle renvoie la liste triée. La syntaxe est la suivante : `liste_triee = sorted(liste)`.
- Par exemple :

```
1 notes = [15,11,10,18,9]
2 note_triees=sort(notes)
3 print(notes_triees)
```

  affichera : `[9,10,11,15,18]`

- On peut obtenir un tri par ordre décroissant en indiquant `reverse=True` `liste_triee = sorted(liste,reverse=True)`.

Trier une liste de dictionnaires

- La fonction `sorted` permet aussi de trier des listes de dictionnaires on indique alors le critère de tri à l'aide de l'option `key`.

Trier une liste de dictionnaires

- La fonction `sorted` permet aussi de trier des listes de dictionnaires on indique alors le critère de tri à l'aide de l'option `key`.
- Par exemple :

```
1 def note(eleve):
2     return eleve["Note"]
3
4 notes = [{"Prenom":"Albert","Note":15},{"Prenom":"Jim","
      Note":10},{"Prenom":"Sarah","Note":19}]
5 notes.sort(key=note,reverse=True)
6 print(notes)
```

affichera : `[{'Prenom': 'Sarah', 'Note': 19}, {'Prenom': 'Albert', 'Note': 15}, {'Prenom': 'Jim', 'Note': 10}]`