Plan

1. chaîne de caractères
   - Qu'est ce qu'une chaîne de caractères
   - Traitement d'une chaîne de caractères en Python
   - Accès aux composantes d'une chaîne de caractères
   - Les fonctions prédéfinies sur les chaînes
2. Les fichiers
   - Généralité sur les fichiers
   - Manipulation des fichiers
   - Les fonctions d'entrées-sorties sur les fichiers textes
   - Le module « pickle »

Définition

Définition

- Une chaîne de caractères en python est **une suite de caractères**.
- On appelle caractère tout élément de l'ensemble :
  - **Alphabet :** 'a','b','c',..,'z' et 'A','B',...'Z'
  - **Signe de ponctuation :** ',',';',' ',' :' '!',' ?'
  - **Caractère spécial :** '/','@','$',...
  - **Chiffres :** '0','1',..,'9'

Exemple

```
>>> nom = "mohamed"
>>> type(nom)
< class 'str' >
```

Les caractéristiques d'une chaîne de caractères en Python

- **Propriété 1**

Une chaîne de caractères est une séquence **ordonnée** et **non modifiable**, chaque élément est classé selon un index numérique et ne peut pas être modifié.

```
>>>ch="Il fait beau"
>>>ch[0]
'I'
>>>ch[0]='m'
Traceback (most recent call last) :
File " < stdin > ", line 1, in< module >
TypeError : 'str' object does not support item assignment
```

- **Propriété 2**

Une chaîne de caractères est dite vide, si elle ne contient aucun caractère, càd que sa longeure est égale à 0

>>> *ch* = "" ou ch=str()

- **Propriété 3**

En, python, le type caractère n'existe pas, par conséquent tout caractère est traité comme un **type str** (si ch='C'alors type(ch) → str)

Déclaration et initialisation d'une chaîne

Syntaxe

identificateur="valeur" ou identificateur='valeur'

- **Exemples**

```
>>>lettre='A'
>>>nom='Ali'
```

Les opérations d'entrée/sortie

lire une chaîne de caractères

- La fonction **input()** permet de lire une chaîne de caractères.
- Exemple :

```
>>>nom=input("donnez votre nom svp :")
```

Afficher une chaîne de caractères

- La fonction **print()** permet d'afficher une chaîne de caractères.
- Exemple :

```
>>>nom="Ali"
>>>print(nom)
'Ali'
```

Les opérateurs : +, * et in

Soient ch1 et ch2 deux chaînes de caractères, avec ch1="salut" et ch2="karim"

- L'opérateur + permet de concaténer les deux chaînes de caractères ch1 et ch2

```
>>>ch1+ch2
'salutkarim'
```

- L'opérateur * permet de dupliquer une chaîne de caractères plusieurs fois.

```
>>>3*ch1
'salutsalutsalut'
```

- x in ch1 : retourne True si x est appartient à la chaîne ch1 et False sinon.

```
>>> 'a' in ch1
True
>>> 'A' in ch1
False
```

- Les deux opérateurs - et / n'existent pas.

Les opérateurs de conversion

On distingue quatre types de conversion :

1. $str --> int$
2. $int --> str$
3. $str --> list$
4. $list --> str$

$str --> int$ et $int --> str$

- $str --> int$

```
>>> S="193"
>>> N=int(S)
>>> N
193
```

- $int --> str$

```
>>> N=310
>>> ch=str(N)
>>> ch
'310'
```

$str --> list$ et $list --> str$

- $str --> list$

```
>>> S="CPGE 2016"
>>> L=list(S)
>>> L
['C','P','G','E',' ','2','0','1','6']
```

- $tuple --> str$

```
>>> ch=['C','P','G','E']
>>> S="".join(ch)
>>> print(S)
CPGE
```

Remarque : une chaîne chiffre est une chaîne contenant uniquement les caractètres chiffres

Parcours chaîne de caractères

Parcours d'une chaîne

L'instruction for est l'instruction idéale pour parcourir une chaîne :

1. Exemple 1 :

```
>>> S="GSR 2016"
>>> for i in range(len(S)) :
        print(S[i],end='')
GSR 2016
```

2. Exemple 2 :

```
>>> S="GSR 2016"
>>> for i in S :
        print(S[i])
G
S
R

2
0
1
6
```

Les fonctions prédéfinies sur les chaînes

Python intègre de nombreuses fonctions qui permettent d'effectuer divers traitements sur les chaînes de caractères. A titre d'exemple, nous citons :

- retourner la taille d'une chaîne : **len**
- comparer deux chaînes de caractères : **cmp**
- calculer le nombre d'occurence d'un caractère : **count**
- conversion de majuscules vers minuscules (resp min vers maj) **upper et lower**
- découpage en chaînes plus petites : **split**
- recherche de mots, *etc.)*.

Les fonctions len, ord, chr et count

- **len(ch)** : cette fonction retourne le nombre de caractères de la chaîne ch

```
>>>ch="maroc"
>>>len(ch)
5
```

- **chr(n)** : affiche un caractère à partir de son code numérique (ASCII) n

```
>>>chr(65)
'A'
```

- **ord("c")** : affiche le code numérique d'un caractère

```
>>>ord('A')
65
```

- **ch.count("c")** : retourne le nombre d'occurence du caractère "c" dans ch

```
>>>ch="Morocco"
>>>ch.count('o')
3
```

Les fonctions de comparaison de caractères et chaînes

- **max(chaine)** renvoie le caractère dont le code numérique est le plus élevé

```
>>> max("maroc")
>>>'r'
```

- **min(chaine)** renvoie le caractère dont le code numérique est le moins élevé

```
>>>min("maroc")
'a'
```

- **max(ch1,ch2)** renvoie la chaîne qui se classe alphabétiquement après l'autre

```
>>> max("Ali","Ahmed")
'Ali'
```

- **min(ch1,ch2)** renvoie la chaîne qui se classe alphabétiquement avant l'autre

```
>>> min("Ali","Ahmed")
'Ahmed'
```

Les fonctions de la mise en forme d'une chaîne

- **ch.upper()** : cette fonction permet de retourner une nouvelle chaîne de caractères contenant la conversion en majuscule de la chaîne ch.

```
>>>ch="Salut les amis" ;ch.upper()
'SALUT LES AMIS'
```

- **ch.lower()** : cette fonction permet de retourner une nouvelle chaîne de caractères contenant la conversion en minuscule de la chaîne ch.

```
>>>ch="CASABLANCA" ;ch.lower()
'casablanca'
```

- **ch.capitalize()** transforme la première lettre de la chaîne ch en capitale

```
>>>ch="maroc 2016" ;ch.capitalize()
'Maroc 2016'
```

- **ch.swapcase()** transforme les minuscules en majuscules et inversement

```
>>>ch="gSr CasA" ;ch.swapcase()()
'GsR cASa'
```

Autres fonctions sur les chaînes de caractères

- **chaine.split()** renvoie une liste de tous les mots (séparation : l'espace) d'une chaîne de caractères.

```
>>>ch="Salut les amis"
>>>ch.split()
['Salut',' les',' amis']
```

- **chaine.split('*')** le séparateur $*$ peut être une chaîne autre que l'espace

```
>>>ch="CASABLANCA"
>>>ch.split('A')
['C',' S',' BL',' NC','']
```

- **chaine.find(ssch)** renvoie la position d'une sous-chaîne dans une chaîne (0 pour la première position, -1 si la sous-chaîne ne s'y trouve pas

```
>>>"Salut les amis".find("les")
6
```

- **chaine.replace(ssch1,ssch2,n)** remplace une sous-chaîne par une autre (éventuellement en limitant à n occurrences)

Plan

1. chaîne de caractères
   - Qu'est ce qu'une chaîne de caractères
   - Traitement d'une chaîne de caractères en Python
   - Accès aux composantes d'une chaîne de caractères
   - Les fonctions prédéfinies sur les chaînes

2. Les fichiers
   - Généralité sur les fichiers
   - Manipulation des fichiers
   - Les fonctions d'entrées-sorties sur les fichiers textes
   - Le module « pickle »

Généralité sur les fichiers

Définition

- Un fichier désigne un ensemble d'informations stocké sur un support physique : disque dur, CD, DVD, clé USB.
- Un fichier se caractérise par :
  1. un nom qui permet de l'identifier de manière unique (unicité) ;
  2. un chemin d'accès qui indique son emplacement dans le disque dur ;
  3. la taille ;
  4. le type.

Types de fichiers

- Généralement, en programmation on distingue deux types de fichiers : :
  1. Fichier texte : les informations sont représentées dans le disque dur selon un code donné(UTF-8, ISO 8859, cp 1250, etc). Ces fichiers sont listables (on peut lire le contenu).
  2. Fichier binaire : les informations (n'importe quel type) sont codées telles qu'elles sont dans la mémoire (brut). Ces fichiers ne sont pas listables.

Ouverture d'un fichier : open(nomFichier,mode,encodage)

Principe

- La fonction **open** permet l'ouvreture d'un fichier.
- L'action d'ouverture signifie deux choses, soit on crée le fichier s'il n'existe pas ou soit on ouvre le fichier pour lire son contenu s'il existe déjà.

Syntaxe de la fonction

- La fonction **open**(NomFichier,mode,encodage) possède **trois** arguments :
  1. NomFichier : c'est la chaîne de caractères qui indique le nom de fichier à ouvrir. En général ce nom, pourra comporter une information (chemin, répertoire...) permettant de préciser l'endroit où se trouve le fichier.
  2. Le mode : c'est la chaîne de caractères qui indique le mode d'ouverture du fichier. Les différents modes les plus utilisés sont les suivants :
     * "r" : lecture seule ;
     * "w" : écriture seule (destruction de l'ancienne version si elle existe) ;
     * "a" : ouvrir le fichier en mode ajout ;
  3. Encodage :cet argument peut être ajouté pour indiquer l'encodage utilisé lors du transfert des données de la RAM vers le disque dur.

La fermeture d'un fichier : close

- La fonction **close**() réalise ce que l'on nomme une fermeture de fichier.
- Cette méthode force l'écriture sur disque dur du tampon associé au fichier

Création d'un fichier

- *Exemple*1
```
>>>f1=open("INFO2.txt","w")
>>> f1.close()
```
- *Exemple*2
```
>>> NomFich=input("Donnez le
nom de fichier à creer :")
>>> f1=open(NomFich,"w")
>>> f1.close()
```

Ouvrir un fichier

- *Exemple*3
```
>>>f1=open("INFO2.txt","r")
>>> f1.close()
```
- *Exemple*4
```
>>> NomFich=input("Donnez le
nom de fichier à ouvrir :")
>>> f1=open(NomFich,"r")
>>> f1.close()
```

Remarques

1. L'utilisation du mode "r" pour l'ouverture d'un fichier en lecture est optionnel ;
2. La précision de l'encodage lors de l'ouvreture est aussi optionnel ;

La fonction d'ecriture : write

Principe

Pour écrire dans un fichier texte, il faut tout d'abord ouvrir ce fichier en mode écriture ou ajout (write 'w' ou append 'a'). L'action de l'écriture consiste à transferer la valeur d'une variable (dans la RAM) de type str vers un fichier existant dans le disque dur.

Syntaxe

La fonction write permet d'écrire des données dans un fichier. Sa syntaxe est : f.write("chaîne de caractères"), f est le descripteur de fichier.

Exemple 1

- Ce code consiste à créer le fichier "test.txt"

```
>>>f=open("test.txt","w")
>>>f.write("salut tout le monde\n")
>>>f.write("Cours INFO2\n")
>>>f.close()
```

.

Résulat 1

- Sur le disque dur "test.txt"

---

salut tout le monde
Cours INFO2

.

La fonction d'ecriture : write

Exemple 2

- Cette fonction permet de sauvegarder la table de multiplication d'un entier n dans un fichier source

```
def TablMulti(source,n) :
    f=open(source,"w")
    for i in range(1,11) :
        f.write(str(i))
        f.write("*")
        f.write(str(n))
        f.write("=")
        f.write(str(i*n))
        f.write("\n")
    f.close()
```

.

Résulat 2

- L'appel de la fonction TablMulti("Table5.txt",5) permet de créer sur le disque dur "Table5.txt"

```
1*5=5
2*5=10
3*5=15
4*5=20
5*5=25
6*5=30
7*5=35
8*5=40
9*5=45
10*5=50
```

.

- Remarque : on peut écrire d="", d=str(i)+"*"+str(n)+"="+str(i*n)+"\n" et f.write(d)

La fonction d'ecriture : write

Ecrire à la fin de fichier

- Il peut être intéressant d'ajouter des lignes à un fichier. Il faut alors l'ouvrir en écriture en mode 'a' et non 'w'.
  >>> *f_out* = open('test.txt', 'a', encoding = 'utf-8')
- Pour rajouter deux lignes à la fin du fichier test.txt on fait :
  >>> *f_out* = open('test.txt', 'a', encoding = 'utf-8')
  >>> *f_out*.write("2014\n")
  >>> *f_out*.write("Pyhton est gratuit\n")
  >>> *f_out*.close()
- Le contenu du fichier "test.txt" devient :

  ---

  salut tout le monde
  Cours INFO2
  2014
  Pyhton est gratuit
- Il n'est pas possible de supprimer des lignes dans un fichier. Il faut créer un nouveau fichier et détruire l'ancien !

Méthodes pour la lecture d'un fichier texte

Principe

Pour lire le contenu d'un fichier texte, il faut tout d'abord ouvrir ce fichier en mode lecture (read 'r'). L'action de lecture consiste à transferer le contenu du fichier (dans le disque dur) vers une variable (dans la RAM). Pour ce faire, on utilise la fonction read :

−− >Les variantes de la fonction read

- read() : la méthode read sans argument lit toutes les données présentes dans le fichier et les transfère dans une variable de type chaîne de caractères (string)

Exemple 1

- Ce code consiste à lire le fichier "test.txt"

```
>>>f=open("test.txt","r")
>>>ch=f.read()
>>>print(ch)
>>>f.close()
```

.

Résulat 1

- Sur l'écran on obtient :

```
salut tout le monde
Cours INFO2
2014
Pyhton est gratuit
```

.

Méthodes pour la lecture d'un fichier texte

- read(size) : la fonction read peut également être utilisée avec un argument size. Celui-ci indiquera combien de caractères doivent être lus, à partir de la position déjà atteinte dans le fichier.

Exemple 2

- Ce code consiste à lire le fichier "test.txt"

```
>>>f=open("test.txt","r")
>>>t=f.read(5)
>>>print(t)
>>>f.close()
```

.

Résulat 2

- Sur l'écran on obtient :
  salut

.

Méthodes pour la lecture d'un fichier texte

- readline() : cette méthode ne lit qu'une seule ligne à la fois à partir du fichier (en incluant le caractère de fin de ligne). Pour lire le contenu entier de fichier, il suffit d'itérer la lecture jusqu'à ce que la ligne soit vide.

Exemple 2

```
>>>f1=open("Table5.txt","r")
>>>ch=f1.readline()
>>>print(ch)
1*5=5
>>>ch=f1.readline()
>>>print(ch)
2*5=10
>>>ch=f1.readline()
>>>print(ch)
3*5=15
>>>f1.close()
```

Parcourir un fichier avec for

```
>>>f=open("test.txt","r")
>>>f
<_io.TextIOWrapper name='test.txt'
encoding='UTF-8'>
>>> for i in f :
    print(i)
salut tout le monde
Cours INFO2
2014
Pyhton est gratuit
>>>f.close()
```

Méthodes pour la lecture d'un fichier texte

Exemple 4 (solution 1)

- Une fonction qui lit un fichier source caractère par caractère

```
def LireFichier(source) :
    f=open(source,"r")
    c=f.read(1)
    while c!="" :
        print(c)
        c=f.read(1)
    f.close()
```

.

Exemple 4 (solution 2)

- Une fonction qui lit un fichier source ligne par ligne

```
def LireFichier(source) :
    f=open(source,"r")
    while 1 :
        lg=f.readline()
        if lg!="" :
          print(lg)
        else :
          break
```

.

Résulat 4 : L'appel LireFichier("test.txt") affiche sur l'écran

```
salut tout le monde
Cours INFO2
2014
Pyhton est gratuit
```

Méthodes pour la lecture d'un fichier texte

- readlines() : cette méthode transfère toutes les lignes d'un fichier dans une liste de chaînes de caractères (chaque élément de la liste est une ligne de fichier).

Exemple 5

```
>>>f=open("test.txt","r")
>>>L=f.readlines()
>>>print(L)
['salut tout le mond\n', 'Cours INFO2\n', '2014\n', 'Python est gratuit\n']
>>>f.close()
```

Remarques

- La méthode readline() est une méthode qui renvoie une chaîne de caractères, alors que la méthode readlines() renvoie une liste ;
- À la fin du fichier, readline() renvoie une chaîne vide, tandis que readlines() renvoie une liste vide.

Le module « pickle »

Les fichiers binaires

- L'encodage binaire utilise une représentation proche de la machine.
- Le module "pickle" permet de traiter des fichiers en mode binaire.
- La méthode dump() permet d'écrire des données de toutes sortes dans un fichier.
- La méthode load() permet de lire des données de toutes sortes dans un fichier.

Exemple

```
>>>from pickle import *
>>>a, b= 27, [12,' CPGE']
>>>f=open('DonneesTest', 'wb')
>>>dump(a,f)
>>>dump(b, f)
>>>f.close()
>>>f= open('DonnesTest', 'rb')
>>>d1,d2= load(f),load(f)
>>>print(d1,d2)
```

Résulat

27, [12,' CPGE']

Remarques

- L'option 'wb' est utilisé pour ouvrir un fichier binaire en écriture
- L'option 'rb' est utilisé pour ouvrir un fichier binaire en lecture