Les Chaines de Caractères

October 25, 2023

1 Les Chaines De Caractères - Généralités

[1]:
```
# Exemples des chaines de caractères
ch1 = 'La formation 2 a commencé'
ch2 = "Aujourd'hui"
ch3 = """Ali a dit : "J'ai commencé mes études"."""
```

[2]:
```
# Déclarer une chaine de caractères vide
ch = ''
# ou
ch = ""
# ou
ch = """"""
# ou
ch = str()
```

[3]:
```
# La fonction len retourne le nombre de caractères d'une chaine de caractères
ch = "Formation 2"
x = len(ch)
print(x)
```

```
11
```

[4]:
```
# Accéder aux caractères d'une chaine de caractères à l'aide des indices␣
 ↪positifs
ch = "Formation 2, abcd123."
print("ch =",ch)
print("ch[0] =", ch[0])
print("ch[1] =", ch[1])
print("ch[9] =", ch[9])
print("ch[11] =", ch[11])
print("ch[17] =", ch[17])
```

```
ch = Formation 2, abcd123.
ch[0] = F
ch[1] = o
ch[9] =
```

```
ch[11] = ,
ch[17] = 1
```

[5]:
```
# Accéder aux caractères d'une chaine de caractères à l'aide des indices␣
 ↪négatifs
ch = "Formation 2, abcd123."
print("ch =",ch)
print("ch[-1] =", ch[-1])
print("ch[-2] =", ch[-2])
print("ch[-9] =", ch[-9])
print("ch[-11] =", ch[-11])
```

```
ch = Formation 2, abcd123.
ch[-1] = .
ch[-2] = 3
ch[-9] =
ch[-11] = 2
```

[6]:
```
# Extraction des sous chaines de caractères (Slicing)
ch = "Formation 2, abcd123."
print("ch =",ch)
print("ch[3 : 7] =", ch[3 : 7])
print("ch[: 5] =", ch[: 5])
print("ch[5 :] =", ch[5 :])
print("ch[3 : 10 : 2] =", ch[3 : 10 : 2])
print("ch[: : 3] =", ch[: : 3])
print("ch[14 : 4 : -1] =", ch[14 : 4 : -1])
print("ch[14 : 4 : -3] =", ch[14 : 4 : -3])
print("ch[: : -1] =", ch[: : -1])
```

```
ch = Formation 2, abcd123.
ch[3 : 7] = mati
ch[: 5] = Forma
ch[5 :] = tion 2, abcd123.
ch[3 : 10 : 2] = mto
ch[: : 3] = Fmi  c2
ch[14 : 4 : -1] = ba ,2 noit
ch[14 : 4 : -3] = b,nt
ch[: : -1] = .321dcba ,2 noitamroF
```

[7]:
```
# La concaténation des chaines de caractères
ch1 = "abcd"
ch2 = "123"
s = ch1 + ch2
print("ch1 =",ch1)
print("ch2 =",ch2)
print("s =",s)
```

```
ch1 = abcd
ch2 = 123
s = abcd123
```

[8]:
```
# La concaténation des chaines de caractères
ch1 = "abcd"
ch2 = "123"
s = ch2 + ch1
print("ch1 =", ch1)
print("ch2 =", ch2)
print("s =",s)
```

```
ch1 = abcd
ch2 = 123
s = 123abcd
```

[9]:
```
# La multiplication d'une chaine de caractères par un entier
ch = "abcd"
p = ch * 5
print("ch =", ch)
print("p =", p)
```

```
ch = abcd
p = abcdabcdabcdabcdabcd
```

[10]:
```
# La multiplication d'une chaine de caractères par un entier
ch = "abcd"
p = 5 * ch
print("ch =", ch)
print("p =", p)
```

```
ch = abcd
p = abcdabcdabcdabcdabcd
```

[11]:
```
# La multiplication d'une chaine de caractères par un entier
ch = 3 * "123"
print("ch =", ch)
```

```
ch = 123123123
```

[12]:
```
# Le test d'appartenance
ch = "Formation 2, abcd123."
ok = 'F' in ch
print(ok)
```

```
True
```

[13]:
```
# Le test d'appartenance
ch = "Formation 2, abcd123."
ok = 'f' in ch
print(ok)
```

```
False
```

[14]:
```
# Le test d'appartenance
ch = "Formation 2, abcd123."
ok = 'tion' in ch
print(ok)
```

```
True
```

[15]:
```
# Le test d'appartenance
ch = "Formation 2, abcd123."
ok = '12' in ch
print(ok)
```

```
True
```

[16]:
```
# Le test d'appartenance
ch = "Formation 2, abcd123."
ok = 'iont' in ch
print(ok)
```

```
False
```

[17]:
```
# Le test d'appartenance
ch = "Formation 2, abcd123."
if 'mat' in ch:
  print("oui")
else:
  print("non")
```

```
oui
```

[18]:
```
# Accéder aux caractères des chaines de caractères à l'aide de la boucle for
ch = "Formation 2"
for c in ch:
  print(c)
```

```
F
o
r
m
a
t
```

```
i
o
n

2
```

[19]:
```
# Accéder aux caractères des chaines de caractères à l'aide de la boucle for
ch = "Formation 2"
for c in ch:
  print(c, end = ' ')
```

```
F o r m a t i o n  2
```

[20]:
```
# Parcourir les indices d'une chaine de caractères à l'aide de la boucle for
ch = "Formation 2"
for i in range(len(ch)):
  print(i)
```

```
0
1
2
3
4
5
6
7
8
9
10
```

[21]:
```
# Parcourir les indices d'une chaine de caractères à l'aide de la boucle for
ch = "Formation 2"
for i in range(len(ch)):
  print(ch[i])
```

```
F
o
r
m
a
t
i
o
n

2
```

[22]:
```
# ATTENTION !!!! LES CHAINES DE CARACTERES SONT NON MUTABLES (NON MODIFIABLES)
# SI ch EST UNE CHAINE DE CARACTERES ET i UN INDICE POSSIBLE DANS ch
# L'ECRITURE ch[i] = 'r' EST NON VALIDE !!!
```

2 Les Chaines De Caractères - Méthodes prédéfinies

- ch.upper() : Elle retoune une copie de la chaine de caractères ch où les alphabets minuscules sont remplacés par leurs majuscules. Les lettres qui étaient déjà en majuscules restent inchangées. ;
- ch.lower() : Elle retoune une copie de la chaine de caractères ch où les alphabets majuscules sont remplacés par leurs miniscules. Les lettres qui étaient déjà en miniscules restent inchangées. ;
- ch.capitalize() : Elle envoie une nouvelle chaîne de caractères avec la première lettre en majuscule et les autres lettres en minuscules. Les lettres qui étaient déjà en majuscules restent inchangées ;
- ch.isnumeric() : Retourne True si ch ne contient que des caractères numériques de 0 à 9 et False sinon ;
- ch.isalpha() : Retourne True si ch ne contient que des caractères qui sont des lettres d'alphabets et False sinon ;
- ch.isalnum() : Retourne True si tous les caractères de ch sont soit alphabétiques soit numériques et False sinon ;
- ch.index(c) : Retourne l'indice de la première occurence de la chaine de caractères c dans ch si c existe dans ch. Sinon la méthode lève une exception (Erreur) ;
- ch.find(c) : Retourne l'indice de la première occurence de la chaine de caractères c dans ch si c existe dans ch. Sinon elle retourne - 1 ;
- ch.count(c) : Retourne le nombre d'occurence de la chaine de caractères c dans ch ;
- ch.split(d) : Retourne une liste de chaines de caractères en divisant la chaine ch par le délimiteur d ;
- d.join(L) : Retourne une chaine de caractères en joignat les chaines qui se trouvent dans la liste L par le délimiteur d.

[23]:
```
# la méthode .upper()
ch = "Bonjour 123"
s = ch.upper()
print(s)
```

```
BONJOUR 123
```

[24]:
```
# la méthode .lower()
ch = "AbdeLJaliL"
s = ch.lower()
print(s)
```

```
abdeljalil
```

[25]:
```
# La méthode .capitalize()
ch = "bonjour, Merci !"
```

```
s = ch.capitalize()
print(s)
```

```
Bonjour, merci !
```

[26]:
```
# La méthode .capitalize()
ch = "Bonjour, Merci !"
s = ch.capitalize()
print(s)
```

```
Bonjour, merci !
```

[27]:
```
# La méthode .capitalize()
ch = "12 Bonjour, Merci !"
s = ch.capitalize()
print(s)
```

```
12 bonjour, merci !
```

[28]:
```
# La méthode .isnumeric()
ch = "1324525241"
test = ch.isnumeric()
print(test)
```

```
True
```

[29]:
```
# La méthode .isnumeric()
ch = "a5e5c41"
test = ch.isnumeric()
print(test)
```

```
False
```

[30]:
```
# La méthode .isnumeric()
ch = "1324525241"
if ch.isnumeric():
  print("numerique")
else:
  print("Non numerique")
```

```
numerique
```

[31]:
```
# La méthode .isnumeric()
ch = "a5e5c41"
if ch.isnumeric():
  print("numerique")
else:
  print("Non numerique")
```

Non numerique

[32]:
```
# La méthode .isalpha()
ch = "a5e5c41"
test = ch.isalpha()
print(test)
```

False

[33]:
```
# La méthode .isalpha()
ch = "sfniozn"
test = ch.isalpha()
print(test)
```

True

[34]:
```
# La méthode .isalpha()
ch = "aabc!"
if ch.isalpha():
  print("Alphabetique")
else:
  print("Non Alphabetique")
```

Non Alphabetique

[35]:
```
# La méthode .isalnum()
ch = "aabc1223!"
if ch.isalnum():
  print("Alphanumerique")
else:
  print("Non Alphanumerique")
```

Non Alphanumerique

[36]:
```
# La méthode .isalnum()
ch = "aabc1223"
if ch.isalnum():
  print("Alphanumerique")
else:
  print("Non Alphanumerique")
```

Alphanumerique

[37]:
```
# La méthode .isalnum()
ch = "aabé1223"
if ch.isalnum():
  print("Alphanumerique")
else:
```

```
  print("Non Alphanumerique")
```

```
Alphanumerique
```

[38]:
```
# la méthode .index()
ch = "Abeljalil Dbira Tlemcani"
i = ch.index('l')
print(i)
```

```
3
```

[39]:
```
# la méthode .index()
ch = "Abeljalil Dbira Tlemcani"
i = ch.index('li')
print(i)
```

```
6
```

[40]:
```
# la méthode .index()
ch = "Abeljalil Dbira Tlemcani"
i = ch.index('ira')
print(i)
```

```
12
```

[41]:
```
# la méthode .find()
ch = "Abeljalil Dbira Tlemcani"
i = ch.find('l')
print(i)
```

```
3
```

[42]:
```
# la méthode .find()
ch = "Abeljalil Dbira Tlemcani"
i = ch.find('lem')
print(i)
```

```
17
```

[43]:
```
# la méthode .find()
ch = "Abeljalil Dbira Tlemcani"
i = ch.find('x')
print(i)
```

```
-1
```

[44]:
```
# La méthode .count()
ch = "Abeljalil Dbira Tlemcani"
```

```
c = ch.count('a')
print(c)
```

```
3
```

[45]:
```
# La méthode .count()
ch = "Abeljalil Dbira Tlemcani"
c = ch.count('x')
print(c)
```

```
0
```

[46]:
```
# La méthode .count()
ch = "Abeljalil Dbira Tlemcani"
c = ch.count('bi')
print(c)
```

```
1
```

[47]:
```
# La méthode .split()
ch = "Abeljalil Dbira Tlemcani"
L = ch.split()
print(L)
```

```
['Abeljalil', 'Dbira', 'Tlemcani']
```

[48]:
```
# La méthode .split()
ch = "Abeljalil Dbira Tlemcani"
L = ch.split('a')
print(L)
```

```
['Abelj', 'lil Dbir', ' Tlemc', 'ni']
```

[49]:
```
# La méthode .split()
ch = "Abeljalil Dbira Tlemcani"
L = ch.split('lil')
print(L)
```

```
['Abelja', ' Dbira Tlemcani']
```

[50]:
```
# La méthode .join()
L = ["Aya", "Rabeh"]
ch = " ".join(L)
print(ch)
```

```
Aya Rabeh
```

[54]:
```
# La méthode .join()
L = ["Aya", "Rabeh"]
ch = "****".join(L)
print(ch)
```

```
Aya****Rabeh
```

3 L'ordre dans les chaines de caractères

Chaque caractère est reconnu par la machine avec un entier unique qui le représente (identifiant). On l'appelle son code ASCII.

- La fonction ord(c) retourne le code ASCII du caractère c ;
- La fonction chr(n) retourne le caractère correspondant au code ASCII n.

[55]:
```
# La fonction ord
print(ord('A'))
```

```
65
```

[56]:
```
# La fonction ord
print(ord('M'))
```

```
77
```

[57]:
```
# La fonction ord
print(ord('?'))
```

```
63
```

[58]:
```
# La fonction chr
print(chr(65))
```

```
A
```

[59]:
```
# La fonction chr
print(chr(77))
```

```
M
```

[60]:
```
# La fonction chr
print(chr(63))
```

```
?
```

Les caractères sont ordonnés selon leur code ASCII

[61]:
```
print(ord('c'))
print(ord('n'))
```

```
print('c' < 'n')
```

```
99
110
True
```

[62]:
```
print(ord('!'))
print(ord('?'))
print('?' < '!')
```

```
33
63
False
```

Les chaînes de caractères sont ordonnées par le code ASCII du premier caractère. Au cas où les premiers caractères sont identiques, on passe au deuxièmes caractères et ainsi de suite. (Similaire à l'ordre des mots dans un dictionnaire).

[63]:
```
print(ord('Z'))
print(ord('M'))
print('Maroc' < 'Zambie')
```

```
90
77
True
```

[64]:
```
print(ord('r'))
print(ord('u'))
print('Maroc' < 'Mauritanie')
```

```
114
117
True
```

[65]:
```
print(ord('m'))
print(ord('M'))
print('maroc' < 'Mauritanie')
```

```
109
77
False
```

4 Exercices

4.1 Exercice 1

Les chaines de caractères sont non modifiables. Ecrire une fonction modifier(ch, i, c) qui prend en paramètre une chaine de caractères ch, un entier i compris entre 0 et len(ch) - 1 et un caractère

c (len(c) = 1). La fonction retourne une nouvelle chaine caractères identique à ch, sauf dans la position (indice) i où on doit avoir le caractère c.

[66]:
```
# Méthode 1
def modifier(ch, i, c):
  p1 = ch[: i]
  p2 = ch[i + 1 :]
  return p1 + c + p2
```

[67]:
```
# Méthode 2
def modifier(ch, i, c):
  return ch[: i] + c + ch[i + 1 :]
```

[68]:
```
ch = "Formation"
chm = modifier(ch, 2, 'R')
print(chm)
```

```
FoRmation
```

[69]:
```
# Méthode 3
def modifier2(ch, i, c):
  p1 = ""
  p2 = ""
  for j in range(i):
    p1 = p1 + ch[j]
  for j in range(i + 1, len(ch)):
    p2 = p2 + ch[j]
  return p1 + c + p2
```

[70]:
```
ch = "Formation"
chm = modifier2(ch, 2, 'R')
print(chm)
```

```
FoRmation
```

4.2 Exercice 2

Ecrire une fonction inverse(ch) qui retourne une copie de la chaine de caractères ch inversée. On doit utiliser deux méthodes.

[71]:
```
# Méthode 1
def inverse(ch):
  return ch[: : -1]
```

[72]:
```
# Méthode 2
def inverse(ch):
  s = ""
  for i in range(len(ch) - 1, -1, -1):
```

```
    s = s + ch[i]
  return s
```

[73]:
```
# Méthode 3
def inverse(ch):
  s = ""
  for x in ch:
    s = x + s
  return s
```

[74]:
```
ch = "Formation"
chi = inverse(ch)
print(chi)
```

```
noitamroF
```

4.3 Exercice 3

Un palindrome est un mot que l'on peut lire indifféremment de gauche à droite et de droite à gauche.

Exemples : radar, aya, tot, kayak.

Ecrire une fonction palindrom(ch) qui prend en parametre une chaine de caractères qui represente un mot tout en miniscule. La fonction retourne True si ch est un palindrome et False sinon.

[75]:
```
# Méthode 1
def palindrom(ch):
  return ch == ch[: : -1]
```

[76]:
```
# Méthode 2
def palindrom(ch):
  return ch == inverse(ch)
```

4.4 Exercice 4

Ecrire une fonction indice(ch, c) qui prend en paramètre une chaine de caractères ch et un caractère c (len(c) = 1) et qui retourne l'indice de la première occurence de c dans ch s'il existe, sinon on retourne -1

[77]:
```
def indice(ch, c):
  for i in range(len(ch)):
    if ch[i] == c:
      return i
  return -1
```

4.5 Exercice 5

Ecrire une fonction majuscule(ch) qui prend en paramètre une chaine de caractères et qui retourne sa version majuscule sans utiliser la méthode .upper(). On ne transformera que les caractères alphabets de a à z ainsi que les caractères alphabets accentués : “éèêëïîùüûàÿç”. Les alphabets accentuées seront transformées en alphabets non accentuées majuscules par exemple le é sera transformé en E.

[78]:
```
# Méthode 1 :
def majuscule(ch):
  min = "abcdefghijklmnopqrstuvwxyzéèêëïîùüûàÿç"
  maj = "ABCDEFGHIJKLMNOPQRSTUVWXYZEEEEIIUUUAYC"
  chm = ""
  for x in ch:
    if x in min:
      i = indice(min, x)
      chm = chm + maj[i]
    else:
      chm = chm + x
  return chm
```

[79]:
```
s = 'l été est fini'
ch = majuscule(s)
print(ch)
```

```
L ETE EST FINI
```

[80]:
```
# Méthode 2
def majuscule2(ch):
  d = ord('a') - ord('A')
  chm = ""
  for c in ch:
    if 'a' <= c <= 'z':
      oc = ord(c)
      oC = ord(c) - d
      chm = chm + chr(oC)
    elif c in "éèêë":
      chm = chm + 'E'
    elif c in "ïî":
      chm = chm + 'I'
    elif c in "ùüû":
      chm = chm + 'U'
    elif c == "à":
      chm = chm + 'A'
    elif c == "ÿ":
      chm = chm + 'Y'
    elif c == "ç":
      chm = chm + 'C'
    else:
```

```
      chm = chm + c
  return chm
```

[81]:
```
s = 'l été est fini'
ch = majuscule2(s)
print(ch)
```

```
L ETE EST FINI
```

4.6 Exercice 6

Ecrire une fonction qui fait la meme chose que la méthode . isnumeric().

[82]:
```
# Méthode 1
def numerique(ch):
  num = "0123456789"
  for x in ch:
    if not x in num:
      return False
  return True
```

[83]:
```
# Méthode 2
def numerique2(ch):
  for x in ch:
    if not ('0' <= x <= '9'):
      return False
  return True
```

4.7 Exercice 7

Ecrire une fonction qui fait la meme chose que la méthode . isalpha(). En ne considère que les alphabets français et les alphabets accentuées suivants “éèêëïîùüûàÿç”.

[84]:
```
# Méthode 1
def alphabetique(ch):
  min = "abcdefghijklmnopqrstuvwxyzéèêëïîùüûàÿç"
  maj = "ABCDEFGHIJKLMNOPQRSTUVWXYZ"
  for x in ch:
    if (not x in min) and (not x in maj):
      return False
  return True
```

[85]:
```
# Méthode 2
def alphabetique2(ch):
  for x in ch:
    if (not ('a' <= x <= 'z')) and (not ('A' <= x <= 'Z')) and (not x in␣
 ↪"éèêëïîùüûàÿç"):
      return False
```

```
  return True
```

4.8 Exercice 8

Ecrire une fonction qui fait la meme chose que la méthode . isalnum(). En ne considère que les alphabets français et les alphabets accentuées suivants “éèêëïîùüûàÿç”.

[86]:
```
def alphanumerique(ch):
  for x in ch:
    if (not numerique(x)) and (not alphabetique(x)):
      return False
  return True
```

4.9 Exercice 9

Ecrire une fonction Indice(ch, c) qui prend en paramètre une chaine de caractères ch et une chaine de caractères c (len(c) >= 1) et qui retourne l’indice de la première occurence de c dans ch s’il existe, sinon on retourne -1

[87]:
```
def Indice(ch, c):
  n = len(c)
  for i in range(len(ch)):
    if ch[i : i + n] == c:
      return i
  return -1
```

4.10 Exercice 10

Ecrire une fonction compter(ch, c) qui prend en paramètre une chaine de caractères ch et une chaine de caractères c (len(c) >= 1) et qui retourne le nombre d’occurence de c dans ch.

[88]:
```
def compter(ch, c):
  n = len(c)
  s = 0
  for i in range(len(ch)):
    if ch[i : i + n] == c:
      s = s + 1
  return s
```

4.11 Exercice 11

Ecrire une fonction diviser(ch, c) qui prend en paramètre une chaine de caractères ch et une chaine de caractères c (len(c) >= 1) et qfait la même chose que ch.split(c).

[89]:
```
# En utilisant les méthodes prédéfinies
def diviser(ch, c):
  L = []
  n = len(c)
```

```
  while c in ch:
    i = ch.index(c)
    L.append(ch[: i])
    ch = ch[i + n :]
  L.append(ch)
  return L
```

[90]:
```
ch = 'abcdefghdeabcder'
L = diviser(ch, 'de')
print(L)
```

```
['abc', 'fgh', 'abc', 'r']
```

[91]:
```
# Sans utiliser les méthode prédéfinies
def diviser2(ch, c):
  L = []
  n = len(c)
  i = 0
  while c in ch:
    if ch[i : i + n] == c:
      L = L + [ch[: i]]
      ch = ch[i + n :]
      i = 0
    else:
      i = i + 1
  L.append(ch)
  return L
```