Entrée [1]:
```
chaine1="ceci est une première chaine"
chaine2='ceci est une deuxième chaine'
print(chaine1)
print(chaine2)
```

```
ceci est une première chaine
ceci est une deuxième chaine
```

Entrée [3]:
```
chaine1="pour gérer l'apostrophe ..."
chaine2='pour gérer les "guillemets" anglais'
print(chaine1)
print(chaine2)

#changer le type de guillemet en fonction de se que tu veux écrire
```

```
pour gérer l'apostrophe ...
pour gérer les "guillemets" anglais
```

Entrée [4]:
```
chaine="Il fait beau pour la saison !"
# La fonction "len" retourne la longueur de la chaîne:
print(len(chaine))
print(chaine)
# On saute une ligne avec un print() "vide"
print()
# On affiche quelques caractères, sachant que le premier caractère a
print(chaine[0])
print(chaine[1])
print(chaine[8])
print()
# Le dernier caractère a le rang correspondant à la (longueur de la c
print(chaine[len(chaine)-1])
```

```
29
Il fait beau pour la saison !

I
l
b

!
```

Entrée [5]:
```
ch1 = "voici le début"
ch2 = "voilà la fin"
ch = ch1 + ch2
print(ch1)
print(ch2)
print(ch)
```

```
voici le début
voilà la fin
voici le débutvoilà la fin
```

Entrée [6]:
```
ch=ch1 + " et " + ch2
print(ch)

#le fait de faire une adiction de variable affecter à une chaine de c
```

```
voici le début et voilà la fin
```

Entrée [8]:
```
# a et b sont créées et initialisées comme des ENTIERS
a=3
b=5
print(a+b)
# a et b sont créées et initialisées comme des CARACTÈRES (ou des cha
a='3'
b='5'
print(a+b)
# la fonction int() transforme une caractère en entier ...
print(int(a)+int(b))
```

```
8
35
8
```

Entrée [9]:
```
# Voici le code ASCII affiché en décimal et en hexadécimal de plusieu
print(f'le code ASCII de A est {ord("A")}, soit {hex(ord("A"))} en he
print(f"le code ASCII de 5 est {ord('5')}, soit {hex(ord('5'))} en he
# 'ord' permet de convertir permet de convertir le code ascii
# et 'hex(ord())' permet de convertir en exa
a='3'
b='Z'
print(f'le code ASCII de {a} est {ord(a)}, soit {hex(ord(a))} en hexa
print(f'le code ASCII de {b} est {ord(b)}, soit {hex(ord(b))} en hexa
```

```
le code ASCII de A est 65, soit 0x41 en hexa
le code ASCII de 5 est 53, soit 0x35 en hexa
le code ASCII de 3 est 51, soit 0x33 en hexa
le code ASCII de Z est 90, soit 0x5a en hexa
```

Entrée [12]:
```
 #La chaîne A est initialisée en tant que "tableau de caractères"
chaineA = 'Tête brûlée'
print(chaineA)
print(f'La longueur de chaineA est: {len(chaineA)}')
print("")
# La chaîne B est initialisée en tant que "tableau d'octets" grâce a
# tout simplement de convertir le TYPE de chaineA de "tableau de car
# sans rien changer au contenu
# A SAVOIR: Comme Python utilise UTF8 par défaut, decode() est équiv
chaineB = chaineA.encode()
print(chaineB)
print(f'La longueur de chaineB est: {len(chaineB)}')
```

```
Tête brûlée
La longueur de chaineA est: 11

b'T\xc3\xaate br\xc3\xbbl\xc3\xa9e'
La longueur de chaineB est: 14
```

Entrée [14]:
```
#Les chaînes A1 et A2 sont initialisées en tant que "tableau de cara

chaineA1 = 'ceci est un message simple'
chaineA2 = 'éàêçèîâûïö'


#Les chaînes B1 et B2 sont initialisées en tant que "tableau d'octet

chaineB1 = chaineA1.encode()
chaineB2 = chaineA2.encode()


print(f'Chaine A1: {chaineA1} de longueur: {len(chaineA1)}')
print(f'Chaine B1: {chaineB1} de longueur: {len(chaineB1)}')
print()
print(f'Chaine A2: {chaineA2} de longueur: {len(chaineA2)}')
print(f'Chaine B2: {chaineB2} de longueur: {len(chaineB2)}')
#on vois ici que si on encode une chaine de charactère spécial, la c
#code hexa
```

```
Chaine A1: ceci est un message simple de longueur: 26
Chaine B1: b'ceci est un message simple' de longueur: 26

Chaine A2: éàêçèîâûïö de longueur: 10
Chaine B2: b'\xc3\xa9\xc3\xa0\xc3\xaa\xc3\xa7\xc3\xa8\xc3\xae\xc3\x
a2\xc3\xbb\xc3\xaf\xc3\xb6' de longueur: 20
```

Entrée [16]:
```
chaineA1 = 'ceci est un message simple'
chaineA2 = 'éàêçèîâûïö'
# Les chaînes B1 et B2 contiendront des codages UTF-8, ... mais c'es
# Le résultat est donc le même qu'avec decode() sans paramètre
chaineB1 = chaineA1.encode('utf8')
chaineB2 = chaineA2.encode('utf8')
print(f'Chaine A1: {chaineA1} de longueur: {len(chaineA1)}')
print(f'Chaine B1: {chaineB1} de longueur: {len(chaineB1)}')
print(f'Chaine A2: {chaineA2} de longueur: {len(chaineA2)}')
print(f'Chaine B2: {chaineB2} de longueur: {len(chaineB2)}')
print()
# Les chaînes B1 et B2 contiendront des codages ISO-8859 (ASCII Lati
# Un caractère = un octet
chaineB1 = chaineA1.encode('8859')
chaineB2 = chaineA2.encode('8859')
print(f'Chaine A1: {chaineA1} de longueur: {len(chaineA1)}')
print(f'Chaine B1: {chaineB1} de longueur: {len(chaineB1)}')
print(f'Chaine A2: {chaineA2} de longueur: {len(chaineA2)}')
print(f'Chaine B2: {chaineB2} de longueur: {len(chaineB2)}')
print()
```

```
Chaine A1: ceci est un message simple de longueur: 26
Chaine B1: b'ceci est un message simple' de longueur: 26
Chaine A2: éàêçèîâûïö de longueur: 10
Chaine B2: b'\xc3\xa9\xc3\xa0\xc3\xaa\xc3\xa7\xc3\xa8\xc3\xae\xc3\x
a2\xc3\xbb\xc3\xaf\xc3\xb6' de longueur: 20

Chaine A1: ceci est un message simple de longueur: 26
Chaine B1: b'ceci est un message simple' de longueur: 26
Chaine A2: éàêçèîâûïö de longueur: 10
Chaine B2: b'\xe9\xe0\xea\xe7\xe8\xee\xe2\xfb\xef\xf6' de longueur:
10
```

Entrée [2]:
```
#initialisation d'une chaine de caractère
chaineA1='ceci est un message simple'
print(chaineA1)

#découpage chaine de caractère "" (espace) = motif de césure
mots=chaineA1.split(" ")
print(mots)

#la  chaine découpe est stocker dans une liste manipulabre comme un t
#on peut comme ci dessous afficher chaque element de la liste 1/1

print()
print(mots[0])
print(mots[1])
print(mots[2])
print(mots[3])
print(mots[4])

#on peut aussi faire une boucle
print()
for mot in mots:
    print(mot)
```

```
ceci est un message simple
['ceci', 'est', 'un', 'message', 'simple']

ceci
est
un
message
simple

ceci
est
un
message
simple
```

Entrée [4]:
```
# Dans cet exemple, on découpe la chaîne selon le motif "e" ...
# Ça ne sert probablement à rien mais ça montre que c'est possible !
# Remarquez quand même que le caractère "e" a disparu de la chaine, c
# en revanche, le caractère "espace" (qui n'est plus le motif de césu

chaineA1 = 'ceci est un message simple'
stupide = chaineA1.split("e")
print(stupide)
# On redécoupe le 3ème élément selon le motif " " (espace)
print(stupide[2].split(" "))
```

```
['c', 'ci ', 'st un m', 'ssag', ' simpl', '']
['st', 'un', 'm']
```

Entrée [5]:
```
chaineA1 = 'ceci est un message simple'
print(chaineA1.split(" ",1))
print(chaineA1.split(" ",2))
print(chaineA1.split(" ",3))
print(chaineA1.split(" ",4))
```

```
['ceci', 'est un message simple']
['ceci', 'est', 'un message simple']
['ceci', 'est', 'un', 'message simple']
['ceci', 'est', 'un', 'message', 'simple']
```

Entrée [6]:
```
chaineA1 = 'ceci est un message simple'
# On affiche les caractères du 10ème au 15ème
# ATTENTION: le premier porte l'index 0 et le dernier n'est jamais a
print(chaineA1[9:15])
# Ne pas mettre le premier index revient à commencer au début
print(chaineA1[:15])

# Ne pas mettre le deuxième index revient à finir à la fin
print(chaineA1[15:])
```

```
un mes
ceci est un mes
sage simple
```

Entrée [7]:
```
chaineA1 = 'ceci est un message simple'

# On affiche les caractères du 5ème au 2ème, mais en comptant depuis
# ATTENTION: comme avec des valeurs positives, le caractère correspo
print(chaineA1[-5:-2])

# Ne pas mettre le premier index revient à commencer au début
print(chaineA1[:-3])

# Ne pas mettre le deuxième index revient à finir à la fin
print(chaineA1[-8:])
```

```
imp
ceci est un message sim
e simple
```

Entrée [ ]:
```

```