Day-4 Strings

Cordes

Le texte est un type de données de chaîne. Tout type de données écrit en texte est une chaîne. Toutes les données sous citation simple, double ou triple sont des chaînes. Il existe différentes méthodes de chaîne et des fonctions intégrées pour gérer les types de données de chaîne. Pour vérifier la longueur d'une chaîne, utilisez la méthode Len ().

Création d'une chaîne

```
letter = 'P'                 # A string could be a single
character or a bunch of texts
print(letter)                # P
print(len(letter))           # 1
greeting = 'Hello, World!'  # String could be made using a
single or double quote,"Hello, World!"
print(greeting)              # Hello, World!
print(len(greeting))         # 13
sentence = "I hope you are enjoying 30 days of Python
Challenge"
print(sentence)
```

La chaîne multiligne est créée en utilisant Triple Single (' ') ou Triple Double Quotes ("" "). Voir l'exemple ci-dessous.

```
Multiline_string = " 'Je suis enseignant et j'aime l'enseignement.  Je n'ai rien trouvé au
ssi gratifiant que les gens qui stimulaient.
C'est pourquoi j'ai créé 30 jours de Python. " 'Print (Multiline_strin
g)

# Une autre façon de faire la même chose Multiline_String = "" "Je suis enseignante e
t j'aime enseigner. Je n'ai rien trouvé aussi gratifiant que les gens habilitants.

C'est pourquoi j'ai créé 30 jours de Python. "" "Print (Multiline_str
ing)
```

Concaténation des cordes

Nous pouvons connecter des chaînes ensemble. La fusion ou la connexion des ch
aînes est appelée concaténation. Voir l'exemple ci-dessous:

```
first_name = 'asabeneh' last_name = 'encoreayeh' espace = " full_na
me = first_name + espace + last_name print (full_name) # asabeneh
yeayeh
```

```
# Vérification de la longueur d'une chaîne à l'aide de Len () Fonction intégrée Print
(Len (First_name)) # 8 PRINT (LEN (LAST_NAME)) # 7 PRINT (LEN (First_Na
me) > Len (Last_Name)) # True Print (Len (Full_Name)) # 16
```

Séquences d'échappement dans les cordes

Dans Python et d'autres langages de programmation \ suivi d'un caractère est une séquence d'évasion. Voyons les personnages d'évasion les plus courants:

- \ n: nouvelle ligne
- \ t: onglet signifie (8 espaces)
- \\: back slash
- \ ': quote unique (')
- \ ": double quote (")

Maintenant, voyons l'utilisation des séquences d'échappement ci-dessus avec des exemples.

```
imprimer ("J'espère que tout le monde apprécie le défi Python. \ Nare vous?") # Line Break

Imprimer ('Days \ Ttopics \ Texercisiciss') # Ajout d'espace d'onglet ou 4 espaces

print ('jour 1 \ t5 \ t5') print ('jour 2 \ t6 \ t20') print ('jour 3 \ t5 \ t23') print ('jour 4 \
t1 \ t35') print ('Ceci est un symbole arrière (\\)') # pour écrire un arrière



print ('dans chaque langage de programmation, il commence par \ "bonjour,
Monde! \ "') # Pour écrire une double citation à l'intérieur d'une seule citation

# sortir
J'espère que tout le monde apprécie le défi Python.  Es-tu ?

JOURS TRICS EXERCICES JOUR
1 5 5 JOUR 2 6 20 JOUR 3 5 23 JOU
R 4 1 35


Ceci est un symbole de barre arrière (\)
Dans chaque langage de programmation, cela commence par "Hello, World!"
```

Formatage des chaînes

Formatage des chaînes à l'ancienne (% opérateur)

À Python, il existe de nombreuses façons de formater les chaînes. Dans cette section, nous en couvrirons certains. L'opérateur "%" est utilisé pour formater un ensemble de variables enfermées dans un "tuple" (une liste de taille fixe), avec une chaîne de format, qui contient du texte normal avec "Spécificateurs d'argument", des symboles spéciaux comme "% s", " % d", "% f", "% .number de digitsf".

- % s - String (ou tout objet avec une représentation de chaîne, comme des nombres)
- % D - entiers
- % F - Nombres de points flottants
- "% .number de Digitsf" - Nombres de points flottants avec précision fixe

```
# Strings uniquement
first_name = 'asabeneh' last_name = 'Language de YEAYEH' = 'python' formed_strin
g = 'Je suis% s% s. J'enseigne% s '% (first_name, last_name, langue) imprimer (formé
d_string)




# Chaînes et nombres rayon = 10 pi = 3.14 zone = pi * rayon ** 2 formé_string = 'La zo
ne de cercle avec un rayon% d est% .2f.' % (rayon, zone) # 2 fait référence aux 2 chiffres
significatifs après le point




python_libraires = ['django', 'flask', 'numpy', 'matplotlib', 'pandas'] formed_string = 'les biblioth
èques python suivantes:% s'% (python_library) imprimer (formé_string) # "suivants sont les bi
bliothèques python 'Numpy', 'Matplotlib', 'Pandas'] "
```

Formatage des chaînes de nouveaux styles (str.format)

Ce formatage est introduit dans Python version 3.

```
first_name = 'asabeneh' last_name = 'encoreayeh' language = 'python' formed_string = 'je suis
{} {}. J'enseigne {} '. Format (First_name, last_name, langue)
```

```
imprimer (formé_string)
a = 4 b =
3

print ('{} + {} = {}'. Format (a, b, a + b)) print ('{} - {} = {}'. print ('{} / {} = {: .2f}'. Form
at (a, b, a / b)) # le limite à deux chiffres après décimal print ('{}% {} = {}'. b, a // b)) print (
'{} ** {} = {}'. Format (a, b, a ** b))




# Sortie 4 + 3 = 7 4
- 3 = 1 4 * 3 = 12 4
/3 = 1,33 4% 3 = 1
4 // 3 = 1 4 ** 3 =
64




# Chaînes et nombres rayon = 10 pi = 3.14 zone = pi * rayon ** 2 forméd_string = 'la zon
e d'un cercle avec un rayon {} est {: .2f}.'.
```

String Interpolation / F-strings (Python 3.6 **+**)

Un autre nouveau formatage de chaînes est l'interpolation de chaîne, les strings F. Les chaînes commencent par F et nous pouvons injecter les données dans leurs positions correspondantes.

```
A = 4 b
= 3
print (f '{a} + {b} = {a + b}') print (f '{a} - {b} =
{a - b}') print (f '{a} * {b} = {a * b}') {b} = {a /
b: .2f} ')

print (f '{a}% {b} = {a% b}') print (f '{a} // {b}
= {a // b}') print (f '{a} ** {b} = {a ** b}')
```

Python Strings comme séquences de personnages

Les chaînes Python sont des séquences de caractères et partagent leurs méthodes d'accès de base avec d'autres séquences d'objets ordonnées Python - listes et tuples. La manière la plu s simple d'extraire des caractères uniques de chaînes (et les membres individuels de n'impo rte quelle séquence) est de les déballer en variables correspondantes.

Déballer les personnages

langue = 'python'

A, B, C, D, E, F = Langue # Déballage des caractères de séquence en variables

Imprimer (a) # p

Imprimer (b) # y

Imprimer (c) # t

imprimer (d) # h

imprimer (e)
o imprimer
(f) # n

Accéder aux caractères dans les chaînes par index

En programmation, le comptage commence à partir de zéro. Par conséquent, la première let tre d'une chaîne est à zéro index et la dernière lettre d'une chaîne est la longueur d'une chaî ne moins une.


```
Langue = 'Python' First_letter = Language [0] PRIN
T (First_letter) # P Second_letter = Language [1] PR
INT (Second_letter) # Y LAST_INDEX = Len (Lan
guage) - 1 Last_letter = Language [Last_index] Print
(Last_letter) # N N
```

Si nous voulons commencer à droite, nous pouvons utiliser l'indexation négative. -1 est le dernier index.

```
Langue = 'Python' Last_letter = Languag
e [-1] print (Last_letter) # n Second_last
= Language [-2]
```

```
print(second_last) # o
```

Trancher les cordes python

Dans Python, nous pouvons trancher des cordes en sous-chaînes.

```
Langue = 'Python' premier_Three = Language [0: 3] # Démarrage à zéro index et jusqu'à 3 mai
s ne pas inclure 3 imprimer (premier_Three) #Pyt Last_Thire = Language [3: 6] Print (Last_Th
ree) # Hon # Another Way Last_There = Language [-3:] Imprime Langue [3:] Imprimer (Last_
There) # Hon
```

Inverser une chaîne

Nous pouvons facilement inverser les cordes en python.

```
salutation = 'Bonjour, monde!'  Imprimer (salutation [:: -
1]) #! Dlrow, Olleh
```

Sauter des personnages tout en tranchant

Il est possible de sauter des caractères tout en tranchant en passant un argument étape p
our trancher la méthode.

```
Langue = 'Python' PTO = Langue [0
: 6: 2] # PRINT (PTO) # PTO
```

Méthodes de cordes

Il existe de nombreuses méthodes de chaîne qui nous permettent de formater les chaînes. V
oir certaines des méthodes de chaîne dans l'exemple suivant:

- Capitalize (): convertit le premier caractère de la chaîne en lettre majuscule

```
Défi = 'Trente jours de Python' Print (Challenge.Capitalize ()) # 'Trente jours de Pyt
hon'
```

- count (): renvoie les occurrences de sous-chaîne dans String, count (substring, start = .., end = ..). Le début est une indexation de départ pour le comptage et la fin est le dernier i ndex à compter.

```
Défi = 'Trente jours de Python' Print (Challenge.Count ('Y'
)) # 3 PRINT (Challenge.Count ('Y', 7, 14)) # 1,

print (Challenge.Count ('th')) # 2 `
```

- Endswith (): vérifie si une chaîne se termine par une fin spécifiée

```
Défi = 'Trente jours de Python' Print (Challenge.endswith ('on
')) # true
print (challenge.endswith ('tion')) # false
```

- expandTabs (): remplace le caractère d'onglet par les espaces, la taille de l'onglet par défaut est 8.

```
Challenge = 'trente \ tdays \ tof \ tpython' imprimer (challenge.expandtabs ()) # 'trente jours de
python' imprimer (challenge.expandtabs (10)) # 'trente jours de



python'
```

- find (): renvoie l'indice de la première occurrence d'une sous-chaîne, si elle n'est pas trouvée Renvoie -1

```
Défi = 'trente jours de Python'
print (challenge.find ('y')) # 5 imprimer (challen
ge.find ('th')) # 0
```

- rfind (): renvoie l'indice de la dernière occurrence d'une sous-chaîne, si elle n'est pas t rouvée renvoie -1

```
Défi = 'Trente jours de Python'
imprimer (défi.rfind ('y')) # 16
imprimer (challenge.rfind ('th')) # 17
```

- format (): formats la chaîne en une plus belle sortie
  sur le formatage des chaînes Vérifiez ce lien ___

```
first_name = 'asabeneh' last_name = 'yetayeh' age = 250 Job = 'professeur' country = 'Finland'
phrase = 'je suis {} {}. Je suis un {}. J'ai {} ans. J'habite dans {}. '. Format (First_name, last_n
ame, âge, travail, pays) imprimer (phrase) # je suis asabeneh encoreayeh. J'ai 250 ans. Je suis e
nseignant. Je vis en Finlande.




rayon = 10 pi = 3.14 zone = pi * ray
on ** 2



résultat = 'La zone d'un cercle avec rayon {} est
{} '. Format (Str (Radius), Str (Area))
```

```
Imprimer (résultat) # La zone d'un cercle avec le rayon 10 est 314
```

- index (): renvoie l'index le plus bas d'une sous-chaîne, les arguments supplémentaires indiquent l'index de démarrage et de fin (par défaut 0 et longueur de chaîne - 1). Si la sous-chaîne n'est pas trouvée, elle soulève une valeur d'énergie.

```
Défi = 'Trente jours de Python' Sub_string = 'Da' Print (Chal
lenge.index (sub_string)) # 7

imprimer (challenge.index (sub_string, 9)) # Erreur
```

- Rindex (): Renvoie l'indice le plus élevé d'une sous-chaîne, les arguments supplémentaires indiquent l'index de démarrage et de fin (par défaut 0 et longueur de chaîne - 1)

```
Challenge = 'trente jours de Python' sub_string = 'da' print (Challenge.
Rindex (sub_string)) # 7 print (Challenge.Rindex (sub_string, 9)) # err
or print (Challenge.Rindex ('on', 8)) # 19
```

- Isalnum (): vérifie le caractère alphanumérique

```
Challenge = 'trentedayspython' imprimer (challeng
e.isalnum ()) # true

Challenge = '30dayspython' imprime (challenge.isa
lnum ()) # true

Défi = 'Trente Days of Python' Print (Challenge.Isalnum ()) # FAUX, l'espace
n'est pas un caractère alphanumérique


Défi = 'Trente jours de Python 2019'
print (challenge.isalnum ()) # false
```

- Isalpha (): vérifie si tous les éléments de chaîne sont des caractères d'alphabet (A-Z et A-Z)

```
Challenge = 'trente jours de Python' Print (Challenge.isalpha ()) # false, l'espace est
une fois de plus exclu le défi = 'trentedayspython' imprimer (challenge.isalpha ()) # t
rue num = '123'
```

- isDecimal (): vérifie si tous les caractères d'une chaîne sont décimaux (0-9)

```
Challenge = 'trente jours de Python' Print (Challenge.isde
cimal ()) # FAUX DIFFIC


Défi = '\ u00b2'                    print (challenge.isdigit ())
# false
```

```
Défi = '12 3 '
imprimer (challenge.isdecimal ()) # false, espace non autorisé
```

- isdigit (): vérifie si tous les caractères d'une chaîne sont des nombres (0-9 et quelques aut res caractères Unicode pour les nombres)

```
Challenge = 'trente' imprimer (challenge.isdigit ()) # f
alse défier = '30' print (challenge.isdigit ()) # true Chal
lenge = '\ u00b2' print (challenge.isdigit ()) # true
```

- iSNumeric (): vérifie si tous les caractères d'une chaîne sont des nombres ou des chiffres (t out comme Isdigit (), accepte juste plus de symboles, comme ½)

```
num = '10' print (num.isnumeric ()) # true num
= '\ u00bd' # ½ print (num.isnumeric ()) # true
num = '10 .5 'print (num.isnumeric ()) # false
```

- iSIdentifier (): vérifie un identifiant valide - il vérifie si une chaîne est un nom de variable valide

```
Challenge = '30daysofpython' imprimer (challenge.isIdentifier ()) # false, car il commenc
e par un numéro de défi = 'trente_days_of_python' imprimer (challenge.isidentifier ()) # t
rue
```

- Islower (): vérifie si tous les caractères de l'alphabet dans la chaîne sont minuscules

```
Défi = 'trente jours de Python'
print (challenge.islolower ()) # True Challenge = 'tre
nte jours de Python'
imprimer (challenge.islolower ()) # false
```

- ISUPper (): vérifie si tous les caractères de l'alphabet dans la chaîne sont en majuscules

```
Défi = 'Trente jours de Python' Print (Challenge.Isupp
er ()) # False Challenge = 'Trente Days of Python' Prin
t (Challenge.isupper ()) # true
```

- join (): renvoie une chaîne concaténée

```
web_tech = ['html', 'css', 'javascript', 'react'] result = '' .join (web_tech) impr
imer (result) # 'html css javascript react' = '#' .join (web_tech) print (result)
# 'html # css # javascript # react'
```

- strip (): supprime tous les caractères donnés à partir du début et de la fin de la chaîne

```
Défi = 'Trente jours de Pythoonnn' Print (Challenge.strip ('Noth')) # 'IRTY D
AYS OF PY'
```

- remplacer (): remplace la sous-chaîne par une chaîne donnée

```
Défi = 'Trente jours de Python' Print (Challenge.replace ('Python', 'Coding')) # 'Trente jours de
codage'
```

- Split (): divise la chaîne, en utilisant une chaîne ou un espace donné comme séparateur

```
Défi = 'Trente jours de Python' Print (Challenge.Split ()) # ['Trente', 'Days', 'of', 'Python'] Chal
lenge = 'Trente, Days, Of, Python' Print (Challenge.Split (',')) # ['Thirty', 'Days', ',', 'Python']
```

- title (): Renvoie une chaîne de titres

```
Défi = 'Trente Days of Python' Print (Challenge.Title ()) # trente jours de
Python
```

- swapcase (): convertit tous les caractères majuscules en minuscules et tous les caractères minuscules en caractères majuscules

```
Challenge = 'Trente Days of Python' Print (Challenge.SwapCase ()) # Trente Day
s of Python Challenge = 'Trente Days of Python' Print (Challenge.Swapcase ()) #
trente jours de Python
```

- startSwith (): vérifie si la chaîne démarre avec la chaîne spécifiée

```
Défi = 'Trente jours de Python' Print (Challenge.startswith ('trente'))
# true

Défi = '30 jours de Python 'Print (Challenge.startswith (' trente ')) # fa
lse
```

Vous êtes une personne extraordinaire et vous avez un potentiel remarquable. Vous venez de terminer les défis du jour 4 et vous êtes à quatre pas de la grandeur vers la grandeur. Faites maintenant quelques exercices pour votre cerveau et vos muscles.

Exercices - Jour 4

1. Concaténer la chaîne «trente», «jours», «de», «python» à une seule chaîne, «trente jours de python». 2. Concaténer la chaîne «codage», «pour», «all» à une seule chaîne, «codage pour tous». 3. Déclarez une variable nommée société et attribuez-la à une valeur initiale " codage pour tous". 4. Imprimez la société variable à l'aide de *print()*. 5. Imprimez la long ueur de la chaîne de l'entreprise en utilisant la méthode *len()* et *print()*. 6. Changez tous le s caractères en lettres majuscules en utilisant la méthode *upper()*. 7. Changez tous les car actères en lettres minuscules en utilisant la méthode *lower()*. 8. Utilisez des méthodes Cap italize (), title (), swapcase () pour formater la valeur de la chaîne *Coding For All*. 9. Cut (tranche) sur le premier mot de la chaîne *Coding For All*. 10. Vérifiez si la chaîne *Coding For All* contient un codage de mots en utilisant l'index de méthode,

trouver ou d'autres méthodes. 11. Remplacez le codage du mot dans la chaîne «codag e pour tous» à Python. 12. Changez Python pour tout le monde à Python pour tous en utili sant la méthode de remplacement ou d'autres méthodes. 13. diviser le codage de la chaîne pour tous les 'en utilisant l'espace comme séparateur (Split ()). 14. "Facebook, Google, Mi crosoft, Apple, IBM, Oracle, Amazon" divisent la chaîne sur la virgule. 15. Quel est le ca ractère à l'index 0 dans la chaîne *Coding For All*. 16. Quel est le dernier index de la chaî ne *Coding For All*. 17. Quel caractère est à l'index 10 dans la chaîne "codage pour toutes ". 18. Créez un acronyme ou une abréviation pour le nom «Python pour tout le monde». 19. Créez un acronyme ou une abréviation pour le nom «codage pour tous».

20. Index d'utilisation pour déterminer la position de la première occurrence de C dans le codage pour tous.

21. Index d'utilisation pour déterminer la position de la première occurrence de F dans le codage pour tous.

22. Utilisez RFIND pour déterminer la position de la dernière occurrence de L dans le cod age de toutes les personnes.

23. Utilisez l'index ou trouvez pour trouver la position de la première occurrence du mot « parce que« dans la phrase suivante: «Vous ne pouvez pas mettre fin à une phrase avec parc e que parce que c'est une conjonction»

24. Utilisez Rindex pour trouver la position de la dernière occurrence du mot car dans la phrase suivante: «Vous ne pouvez pas mettre fin à une phrase avec parce que parce que c'est une conjonction»

25. Tranchez l'expression «parce que parce que« dans la phrase suivante: «Vous ne pouve z pas mettre fin à une phrase avec parce que parce que parce que c'est une conjonction»

26. Trouvez la position de la première occurrence du mot «parce que« dans la phrase suiv ante: «Vous ne pouvez pas mettre fin à une phrase avec parce que parce que c'est une con jonction»

27. Trancher l'expression «parce que parce que« dans la phrase suivante: «Vous ne pouve z pas mettre fin à une phrase avec parce que parce que c'est une conjonction»

28. Le codage pour tous les 'commence-t-il par une sous-chaîne *Coding*?

29. Le codage pour tous les 'se termine-t-il par une sous-chaîne *coding*?

30. 'Codage pour tous', retirez les espaces de fuite gauche et droit dans la chaîne donnée .

31. Laquelle des variables suivantes renvoie vrai lorsque nous utilisons la méthode iSi dentifier ():

o 30daysofpython o Thirty_d
ays_of_python

32. La liste suivante contient les noms de certaines bibliothèques Python: [«Django», «Fla sk», «Bottle», «Pyramid», «Falcon»]. Rejoignez la liste avec un hachage avec Space Strin g.

33. Utilisez la nouvelle séquence d'échappement de ligne pour séparer les phrases suivantes.

```
J'apprécie ce défi.  Je me demande juste quell
e est la prochaine étape.
```

34. Utilisez une séquence d'échappement d'onglet pour écrire les lignes suivantes.

```
Name       Age      Country   City
Asabeneh   250      Finland   Helsinki
```

35. Utilisez la méthode de formatage de chaîne pour afficher ce qui suit:

```
RADIUS = 10 Zone = 3.14 * RADIUS ** 2 La zone d'un cercle avec le rayon 10 est de
314 mètres carrés.
```

36. Faites ce qui suit en utilisant les méthodes de formatage des chaînes:

```
8 + 6 = 14
```

```
8 - 6 = 2 8 * 6 = 48 8/6
= 1,33 8% 6 = 2 8 // 6 =
1 8 ** 6 = 262144
```

Félicitations!