exercices_PDF_17_Date_24_08_2020

August 26, 2020

[3]:
```
def polynome ( x ) :
    x2 = x*x
    return x2 + x - 5
polynome(10)
```

[3]:
```
105
```

[5]:
```
from math import log# on importe une fonction existante
def log_base ( x, base = 10) :
    return log (x) / log(base)
y = log_base (1000)    # identique à y = log_base (1000, 10)
z = log_base (1000, 2) # logarithme en base deux
y,z
```

[5]:
```
(2.9999999999999996, 9.965784284662087)
```

[6]:
```
def calcul(x) :
    y = x**2
    z = x + y
    return a

print(a) # déclenche une exception
```

```
␣
↪---------------------------------------------------------------------------

NameError                                 Traceback (most recent call␣
↪last)

<ipython-input-6-27ed1f968f03> in <module>
  4     return a
  5
----> 6 print(a) # déclenche une exception


NameError: name 'a' is not defined
```

[9]:
```
a = 2

def test() :
    global a
    a = 3
    return print("intérieur de test", a)

test()
print("après test", a)
```

```
intérieur de test 3
après test 3
```

[12]:
```
def recursive (x) :
    if x / 2 < 1 :
        print("je m'appelle pas pour x=", x)
        return 1
    else :
        print("je m'appelle pour x=",x)
        return recursive (x/2) + 1

recursive (10)
```

```
je m'appelle pour x= 10
je m'appelle pour x= 5.0
je m'appelle pour x= 2.5
je m'appelle pas pour x= 1.25
```

[12]:
```
4
```

[19]:
```
s1 = "Du passe faisons table rase"
l1 = [i for i in s1]
l2 = s1.split("as")
l3 = s1.split()

print(l1[0], l1[1], l1[2], l1[3])
print(l2)
print(l3)
```

```
D u   p
['Du p', 'se faisons table r', 'e']
['Du', 'passe', 'faisons', 'table', 'rase']
```

[21]:
```
l1 = ['H', 'e', 'l', 'l', 'o', ' ', 'W']
l2 = ['Du', 'passe', 'faisons', 'table', 'rase']
s1 = "".join(l1)
s2 = " ".join(l2)
```

```
print(s1)
print(s2)
```

```
Hello W
Du passe faisons table rase
```

[22]:
```
s1 = "Faisons table rase"
s2 = s1[:8].lower() + s1[8:13].upper()
print(s2)

# la méthode upper() transforme les lettres en majuscules
# la méthode() les passe en minuscules
```

```
faisons TABLE
```

[23]:
```
s1 = "Ce texte contient deux fois le mot texte."
print(s1.find('texte', 0))
print(s1.find('texte', 4))
print(s1.find('texte', 36))

# Ici on a utiliser la mèthode find(x, debut, fin) qui permet de rechercher un␣
 ↪terme x
# dans ma chaîne à la quel on applique cette mèthode.
# On spécifiant l'index de début et l'index de fin qui définisse une plage de␣
 ↪recherche dans cette chaîne.
```

```
3
35
-1
```

[24]:
```
s1 = "Ce texte contient deux fois le mot texte."
max = 0
while max > -1:
    max = s1.find('texte', max+1)
    print(max)
```

```
3
35
-1
```

[29]:
```
s1 = "Le prix est de deux euros."
# if 'euros' in s1:
i  = s1.index('euros')
s2 = s1[:i-5] + '$' + s1[i-5:i]
j  = s2.find('deux')
s3 = s2[:j] + '2;15' + s2[j+4:]
print(s3)
```

```
Le prix est de $2;15
```

[26]:
```
s1 = "Le prix est de deux euros."
print(s1[:i-5])
```

```
Le prix est de
```

[27]:
```
s1 = "Le prix est de deux euros."
print(s1[i-5:i])
```

```
deux
```

[32]:
```
s1 = "Le prix est de deux euros."
s2 = s1.replace('deux euros', '$2.15')
print(s2)
```

```
Le prix est de $2.15.
```

[33]:
```
s = '300 seulement ?'
l = s.split()
for mot in l:
    if mot.isalpha(): # isalpha() qui test si une chaîne est composé uniquement␣
 ↪de lettres.
        print('mot')
    if mot.isnumeric(): # isnumeric() qui test si une chaîne est componsé␣
 ↪uniquement de chiffres.
        print('nombre')
    if not mot.isalnum(): # isalnum() qui test si une chaîne est composé d'un␣
 ↪mélange des deux.
        print('?')
```

```
nombre
mot
?
```

[39]:
```
from re import findall
s = "Le PIB de l'Argentine baisse depuis 3 ans"
l1 = findall('[A-Z][a-z]+', s)
l2 = findall('[a-z A-Z]*[iI][a-z A-Z]*', s)
print(l1)
print(l2)
```

```
['Le', 'Argentine']
['Le PIB de l', 'Argentine baisse depuis ']
```

[48]:
```
#from re import findall
motif = '0[1-9](?:[\s\.]?[0-9]{2}){4}'
n1 = "0678828383,"
```

```
n2 = "09.34.67.12.11,"
n3 = "03 11 23 20 38,"
n4 = "03 11 23 20,"
n5 = "03.11 23 2038,"
n6 = "03-23-20-20-38,"
s = n1+n2+n3+n4+n5+n6
print(findall(motif, s))
```

```
['0678828383', '09.34.67.12.11', '03 11 23 20 38', '03.11 23 2038']
```

[49]:
```
from re import sub
s = 'Un texte <strong>HTML<strong/>avec des balises'
s += ' et même<script type="text/javascript">'
s += 'var i = 5 ;</script> du javascript dedans.'
s1 = sub('<[a-z]*>', '', s)
print(s1)
```

```
Un texte HTML<strong/>avec des balises et même<script type="text/javascript">var
i = 5 ;</script> du javascript dedans.
```

[52]:
```
#from re import sub
s = 'Un texte <strong>HTML<strong/> avec des balises'
s += ' et même<script type="text/javascript">'
s += ' var i = 5 ;</script> du javascript dedans.'
s1 = sub('<.*>', '', s)
s2 = sub('<[a-z\\/"=\s]*>', '', s)
s3 = sub('<[^>]*>', '', s)
print(s1)
print(s2)
print(s3)
```

```
Un texte  du javascript dedans.
Un texte HTML avec des balises et même var i = 5 ; du javascript dedans.
Un texte HTML avec des balises et même var i = 5 ; du javascript dedans.
```

[114]:
```
a = 2

def test() :
    global a
    a = 3
    return print("intérieur de test", a)

test()
print("après test", a)
```

```
intérieur de test 3
après test 3
```

[63]:
```
###Exercice:
# 1 - Ecrire une fonction hascap(s) qui renvoie tous les mots de
#la chaîne s commençant par une majuscule.
#Pour se faire utiliser la fonction ord() pour obtenir le code
# ASCII des lettres
#(Les lettres majuscules ont un code allant de 65 à 90).

a = "Alexandre Beuil simplon sur Cannes"
b = a.split()

def hascap(c):
    test = []
    for i in c :
        if ord(i[0]) >= 65 and ord(i[0]) <= 90 :
            test.append(i)
    return test

hascap(b)
```

[63]:
```
['Alexandre', 'Beuil', 'Cannes']
```

[97]:
```
# 2 - Proposer une fonction inflation(s) qui va doubler la valeur
# de tous les nombres dans la chaîne s.
#Exemple:"Le prix est de 27 euros" devient "Le prix est de
# 54 euros".
#Utiliser la fonction enumerate() pour lancer une boucle for
# (Taper dans Google"enumerate boucle for".)


liste1 = "Le prix est de 27 euros"
liste2 = liste1.split()

def inflation(liste2):
    for x,mot in enumerate(liste2) : # enumerate renvoie des ligne␣
 ↪indice,element de la liste
        if mot.isnumeric():
            double = int(mot)*2

    return print( "Le prix est de {} euros".format(double))

inflation(liste2)


#s = "Le prix est de 27 euros"
#def inflation(s):

#    split_s = s.split()
```

```
#    print(split_s)
#    for i,j in enumerate(split_s):
#        if j.isnumeric():
#            split_s[i] = str(int(j)*2)
#    result = " ".join(split_s)
#    return print(result)

#inflation("j'ai 30 ans")
```

```
Le prix est de 54 euros
```

[80]:
```
# 3 - Proposer une fonction lignes qui à partir d'une long chaîne s
# (<100 caractères) renvoie une liste de chaîne de caractères
# contenant chacun 24 caractères maximum et terminant par un espace.

# Voici un exemple de chaîne sur le quel vous pouvez travailler :
# s = "Onze ans déjà que cela passe vite Vous"
# s += "vous étiez servis simplement de vos armes la"
# s += "mort n'éblouit pas les yeux des partisans Vous"
# s += "aviez vos portraits sur les murs de nos villes"

#import re

s  = "Onze ans déjà que cela passe vite Vous "
s += "vous étiez servis simplement de vos armes la "
s += "mort n'éblouit pas les yeux des partisans Vous "
s += "aviez vos portraits sur les murs de nos villes "

l = []
l2 = ""
T = s.split()
for mot in T:
    if len(mot) +1 + len(l2) >24:
        l.append(l2)
        l2 = mot + " "
    else:
        l2+=mot + " "

l.append(l2)
print(l)


#def test(p):
#    print(re.findall('.{0,24} ', p))

#test(s)
```

```
['Onze ans déjà que cela ', 'passe vite Vousvous ', 'étiez servis simplement ',
```

```
'de vos armes lamort ', "n'éblouit pas les yeux ", 'des partisans Vousaviez ',
'vos portraits sur les ', 'murs de nos villes ']
```

[122]:
```python
# 4 - Proposer un programme qui renvoie la liste de tout les nombres
# (y compris décimaux et négatifs) d'une chaîne de caractères.
# A tester sur la chaîne : << Les 2 marquereaux valent 6.50 euros>>.

import re

a  = "Les 2 marquereaux valent 6.50 euros."
x1 = findall('[0-9]+\.*[0-9]*', a)
print(x1)
```

```
['2', '6.50']
```

[126]:
```python
# 5 - Proposer une fonction arrondi(s) qui dans la chaîne s
# troncature tout les nombre décimaux. On autorise les nombres
# négatifs.
# Pour ce faire, vous avez la pôssibilité d'utiliser :
# - des () pour désigner des blocs de données dans l'expression
# - rationnelle.
# - pour remplacer chacun des blocs l'expression est r'\1_\2_'.


import re

p = "il a 8.45 et 20.43 euros sur son compte "

def test(s):
    s= re.sub("[.,][0-9]+", "",s)
    print(s)


test(p)
```

```
il a 8 et 20 euros sur son compte
```

[ ]:
```python

```