TDO-seance1_exos

September 20, 2018

1 L3 E - S5 2018-19- TDO - séance 1, solutions des exercices

2 TDO 1 - Exercices

2.1 Exercice 1 - Conversion de secondes

1. Ecrire un programme qui convertit en années, mois, jours, heures, minutes et secondes une durée de 123456789 secondes et afficher le résultat sous la forme 123456789 secondes = a années j jours h heures m minutes s secondes

2. Ecrire un programme qui convertit en millénaires, siècles, années, mois, jours, heures, minutes et secondes une durée de 123456789987654321 secondes et afficher le résultat sous la forme 123456789987654321 secondes = M millénaires S siècles a années j jours h heures m minutes s secondes

In [1]:
```
n=123456789
s=n%60
n1=(n-s)//60
m= n1% 60
n2=(n1-m)//60
h=n2 % 24
n3=(n2-h)//24
j=n3 % 365
a=(n3-j)//365
print("on ote",s, "secondes et il reste", n-s, "secondes" )
print("on ote",m, "minutes et il reste", n-s-60*m, "secondes" )
print("on ote",h, "heures et il reste", n-s-60*m-3600*h, "secondes" )
print("on ote",j, "jours et il reste", n-s-60*m-3600*h-24*3600*j, "secondes" )
print("on ote",a, "années et il reste", n-s-60*m-3600*h-24*3600*j-365*3600*24*a, "second
print("\n", n, "secondes = ", a, "années", j, "jours", h, "heures", m, "minutes", s, "se
```

```
on ote 9 secondes et il reste 123456780 secondes
on ote 33 minutes et il reste 123454800 secondes
on ote 21 heures et il reste 123379200 secondes
on ote 333 jours et il reste 94608000 secondes
on ote 3 années et il reste 0 secondes

 123456789 secondes =  3 années 333 jours 21 heures 33 minutes 9 secondes
```

```
In [2]: n=123456789
        s=n%60
        m=n//60 %60
        h=n//3600 %24
        j=n//(24*3600) %365
        a=n//(24*3600*365)

        print("\n", n, "secondes = ", a, "années", j, "jours", h, "heures", m, "minutes", s, "se
```

```
 123456789 secondes =  3 années 333 jours 21 heures 33 minutes 9 secondes
```

2.1.1 Une version plus rapide :

```
In [3]: n=123456789987654321

        s=n%60
        n1=(n-s)//60
        m= n1% 60
        n2=(n1-m)//60
        h=n2 % 24
        n3=(n2-h)//24
        j=n3 % 365
        n4=(n3-j)//365
        a=n4%100
        n5=(n4-a)//100
        S=n5%10
        M=(n5-S)//10
        print(s)
        print("il reste", n-s, "secondes" )
        print(m)
        print("il reste", n-s-60*m, "secondes" )
        print(h)
        print("il reste", n-s-60*m-3600*h, "secondes" )
        print(j)
        print("il reste", n-s-60*m-3600*h-24*3600*j, "secondes" )
        print(a)
        print("il reste", n-s-60*m-3600*h-24*3600*j-365*3600*24*a, "secondes" )
        print(S)
        print("il reste", n-s-60*m-3600*h-24*3600*j-365*3600*24*a-100*365*3600*24*S, "secondes"
        print(M)
        print("il reste", n-s-60*m-3600*h-24*3600*j-365*3600*24*a-100*365*3600*24*S-1000*365*360
        print("\n", n, "secondes = ", M, "millénaires", S, "siècles", a, "années", j, "jours", h
```

```
21
il reste 123456789987654300 secondes
25
```

```
il reste 123456789987652800 secondes
12
il reste 123456789987609600 secondes
179
il reste 123456789972144000 secondes
29
il reste 123456789057600000 secondes
1
il reste 123456785904000000 secondes
3914789
il reste 0 secondes

 123456789987654321 secondes =  3914789 millénaires 1 siècles 29 années 179 jours 12 heures 25 m
```

2.1.2 Et la version plus rapide :

```
In [4]: n=123456789987654321
        s=n%60
        m=n//60 %60
        h=n//3600 %24
        j=n//(24*3600) %365
        a=n//(24*3600*365) %100
        S=n//(24*3600*365*100) %10
        M=n//(24*3600*365*1000)
        print("\n", n, "secondes = ", M, "millénaires", S, "siècles", a, "années", j, "jours", h

 123456789987654321 secondes =  3914789 millénaires 1 siècles 29 années 179 jours 12 heures 25 m
```

2.2 Exercice 2 - Conversion de degrés en radians

Ecrire un programme qui covertit en radians un angle A de 55 degrés 36 minutes et 28 secondes en affichant le résultat sous la forme : un angle de 155 ° 8 ′ 13 " vaut x radians

```
In [5]: from math import *
        deg,min,sec=155,8,13
        angle_en_degres=deg+min/60+sec/3600
        angle_en_radians= angle_en_degres * pi /180
        print("Un angle de ",deg,"°",min,"'",sec,"\" vaut" , angle_en_radians, " radians")

Un angle de  155 ° 8 ' 13 " vaut 2.7076504720390804  radians
```

2.3 Exercice 3 - Alphabets

a) A partir de la chaîne 'abcdefghijklmnopqrstuvwxyz' (qu'on appellera alpha), utiliser la boucle while pour construire une chaîne 'a b c d e f g h i j k l m n o p q r s t u v w x y z' (qu'on appellera nouvel_alpha).

b) Recommencer le même exercice avec la chaîne 'saperlipopette'.

2.3.1 a) nouvel alphabet

```
In [6]: alpha='abcdefghijklmnopqrstuvwxyz'
        i=0
        nouvel_alpha=""
        while i<26:
            t=alpha[i]
            nouvel_alpha=nouvel_alpha+t+' '
            i=i+1
        nouvel_alpha
        print('nouvel_alpha =', nouvel_alpha)
```

```
nouvel_alpha = a b c d e f g h i j k l m n o p q r s t u v w x y z
```

```
In [7]: alpha='Telmo fieux'
        i=0
        nouvel_alpha=""
        while i<11:
            t=alpha[i]
            nouvel_alpha=nouvel_alpha+t+' '
            i=i+1
        nouvel_alpha
        print('nouvel_alpha =', nouvel_alpha)
```

```
nouvel_alpha = T e l m o   f i e u x
```

2.3.2 nouveau saperlipopette

```
In [8]: saper='saperlipopette'
        i=0
        nouveau_saper=""
        while i<len(saper):
            t=saper[i]
            nouveau_saper=nouveau_saper+t+' '
            i=i+1
        nouveau_saper
        print('nouveau_saper =', nouveau_saper)
```

```
nouveau_saper = s a p e r l i p o p e t t e
```

2.4 Exercice 4 - Tables de multiplication

a) Utiliser la boucle while pour afficher la table de multiplication de 6 sous la forme :

1 x 6 = 6 2 x 6 = 12 3 x 6 = 18 etc...
jusqu'à 10 x 6 =60

b) Sur le même modèle, utiliser deux boucles while pour afficher les tables de multiplication de 2 à 12 ; les tables de multiplications de deux entiers successifs seront séparées par une ligne vide.

2.4.1 a) la table de multiplication de 6

```
In [9]: m=1
        while m < 11:
            print(m, "x 6 =", 6*m)
            m = m +1
```

```
1 x 6 = 6
2 x 6 = 12
3 x 6 = 18
4 x 6 = 24
5 x 6 = 30
6 x 6 = 36
7 x 6 = 42
8 x 6 = 48
9 x 6 = 54
10 x 6 = 60
```

2.4.2 b) les tables de multiplication de 2 à 12

```
In [10]: n=2
         while n<=12:
             m=1
             while m < 11:
                 print(m, "x", n," =", n*m)
                 m = m +1
             print('\n')
             n=n+1
```

```
1 x 2  = 2
2 x 2  = 4
3 x 2  = 6
4 x 2  = 8
5 x 2  = 10
6 x 2  = 12
7 x 2  = 14
8 x 2  = 16
9 x 2  = 18
10 x 2  = 20
```

1 x 3 = 3
2 x 3 = 6
3 x 3 = 9
4 x 3 = 12
5 x 3 = 15
6 x 3 = 18
7 x 3 = 21
8 x 3 = 24
9 x 3 = 27
10 x 3 = 30

1 x 4 = 4
2 x 4 = 8
3 x 4 = 12
4 x 4 = 16
5 x 4 = 20
6 x 4 = 24
7 x 4 = 28
8 x 4 = 32
9 x 4 = 36
10 x 4 = 40

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

1 x 6 = 6
2 x 6 = 12
3 x 6 = 18
4 x 6 = 24
5 x 6 = 30
6 x 6 = 36
7 x 6 = 42
8 x 6 = 48
9 x 6 = 54
10 x 6 = 60

```
1 x 7  = 7
2 x 7  = 14
3 x 7  = 21
4 x 7  = 28
5 x 7  = 35
6 x 7  = 42
7 x 7  = 49
8 x 7  = 56
9 x 7  = 63
10 x 7  = 70


1 x 8  = 8
2 x 8  = 16
3 x 8  = 24
4 x 8  = 32
5 x 8  = 40
6 x 8  = 48
7 x 8  = 56
8 x 8  = 64
9 x 8  = 72
10 x 8  = 80


1 x 9  = 9
2 x 9  = 18
3 x 9  = 27
4 x 9  = 36
5 x 9  = 45
6 x 9  = 54
7 x 9  = 63
8 x 9  = 72
9 x 9  = 81
10 x 9  = 90


1 x 10  = 10
2 x 10  = 20
3 x 10  = 30
4 x 10  = 40
5 x 10  = 50
6 x 10  = 60
7 x 10  = 70
8 x 10  = 80
9 x 10  = 90
10 x 10  = 100
```

```
1 x 11  = 11
2 x 11  = 22
3 x 11  = 33
4 x 11  = 44
5 x 11  = 55
6 x 11  = 66
7 x 11  = 77
8 x 11  = 88
9 x 11  = 99
10 x 11  = 110
```

```
1 x 12  = 12
2 x 12  = 24
3 x 12  = 36
4 x 12  = 48
5 x 12  = 60
6 x 12  = 72
7 x 12  = 84
8 x 12  = 96
9 x 12  = 108
10 x 12  = 120
```

2.5 Exercice 5 - La plus petite puissance négative de 2

Utiliser la condition while pour trouver le plus petit réel strictement positif de la forme $1/(2^n)$ et indiquer pour quel entier n il est atteint.

```
In [11]: x=1
         n=0
         while x >0.00:
             x=x/2
             n=n+1
         print("la plus petite puissance négative de 2 est",1/(2**(n-1)),", obtenue pour n=", n-
```

```
la plus petite puissance négative de 2 est 5e-324 , obtenue pour n= 1074
```

2.6 Exercice 6 - Les nombres de Fibonacci

On rappelle que la suite de Fibonacci est définie par $F(0) = F(1) = 1$ et $F(n+2) = F(n) + F(n+1)$.

a) Utiliser la boucle `while` pour calculer $F(100)$.

b) Trouver le plus grand entier n tel que $F(n) < 10^{20}$ puis afficher n et $F(n)$.

```
In [1]: a,b=1,1
        n=0
        while n <10:
            a,b=b,a+b
            n=n+1
            print(a)
        print("F(",n,")=",a)
```

```
1
2
3
5
8
13
21
34
55
89
F( 10 )= 89
```

```
In [13]: a,b=1,1
         n=0
         while b<10**(20):
             a,b=b,a+b
             n=n+1
             #print(a)
         print("Le plus grand entier n tel que F(n)<10^(20) est", n,"et F(",n,")=",a )
```

```
Le plus grand entier n tel que F(n)<10^(20) est 96 et F( 96 )= 83621143489848422977
```

2.7 Exercice 7 - Calcul du pgcd

Pour obtenir le PGCD de a et de b, on effectue la division euclidienne $a = b * q_1 + r_1$, puis $b = r_1 * q_2 + r_2$, puis $r_1 = r_2 * q_2 + r_3$, etc... Le dernier reste non nul dans cette suite de divisions est le PGCDde a et b. Utiliser l'instruction `while` pour calculer le PGCD de 573804 et 348096 à l'aide de cette méthode.

```
In [14]: a,b=573804, 348096
         while b!=0:
             a,b=b,a%b
         print(a)
```

```
84
```

```
In [ ]:
```

```
In [ ]:
```