Nom : **Prénom :**

TD : Algorithmiques Suites

Exercice 1 : Termes d'une suite définie par une relation de récurrence

```python
def suite(N):
    u=500
    n=0
    while n<=N:
        n=n+1
        u=0.9*u+1
    return n,u
```

1) Compléter le tableau suivant afin d'étudier la valeur des variables n et u pour chaque itération si on choisit N=5

u	500	451	406,9	367,21	331,489	299,3401	270,40609
n	0	1	2	3	4	5	6
Condition $n \leq N = 5$	VRAIE	VRAIE	VRAIE	VRAIE	VRAIE	VRAIE	FAUX

Donc la fonction va renvoyer : 6 ; 270,40609 si on écrit suite(5) dans la console. Vérifier-le.

2) Expliciter la suite associée à cet algorithme. Pour n entier naturel, : $\begin{cases} u_0=500 \\ u_{n+1}=0{,}9\,u_n+1 \end{cases}$

3) On remarque que ce programme ne renvoie pas le terme de rang 5 mais le suivant. Expliquer quelle modification apporter pour corriger ce problème. la condition n<N

4) Comment modifier ce programme afin d'afficher les termes de la suite de rang 1 à N (**avec les rangs associés**) . Vous devez utiliser une boucle while

On remplace le return n,u par print n,u et on le place dans la boucle while

5) Ecrire un autre programme qui affichera les termes de la suite de rang 0 à N (**avec les rangs associés**) en utilisant cette fois une boucle "for i in range ... "

```
def suite(n) :
    u=500
    n=0
    for i in range(n) :
        print(i,u)
        u=0,9*u+1
```

6) Conjecturer la limite quand n tend vers +∞ de la suite (u_n) à l'aide de vos algorithmes en calculant les termes d'indice 10 , 20 , 50 , 100 par exemple.

u_{10} = 180,85 u_{20} = 69,57 u_{50} = 12,52 u_{100} = 10,01

Il semble que la suite (u_n) tende vers : 10 on a u_{1000} = 10,00000

<u>Exercice 2 :</u> Problème de seuil

Un couple fait un placement au taux annuel de 2 % dont les intérêts sont capitalisés tous les ans. Le couple a placé le montant de 500 euros à l'ouverture le 1er janvier 2010 puis, tous les ans à chaque 1er janvier, verse 1 200 euros.
On note u_n le capital présent sur le compte le 1er janvier 2010+n après le versement annuel.

On a donc u_0 = 500 et on admet que pour tout n, la suite (u_n) est définie par : $\begin{cases} u_0=500 \\ u_{n+1}=1{,}02\times u_n+1200 \end{cases}$

1) L'algorithme suivant permet d'afficher la somme présente sur le compte après un certain nombre d'années.
Compléter le et tester le programme avec quelques valeurs de n.
u = 1.02*u+1200

```
def U(N):
    u,n=500,0
    while n<N:
        n=n+1
        u=...
    return u
```

2) Problème de seuil.

L'objectif du couple est de constituer un capital de 18 000 euros.
On cherche donc maintenant à écrire un algorithme qui renvoie l'indice et la valeur du premier terme de la suite qui dépasse 18 000.
Compléter et tester le programme en déterminant l'année durant laquelle le couple aura atteint son objectif.

Objectif atteint en : 2010+13=2023

l'affichage est 18263,2 pour n = 23

```
def U(N):
    u,n=500,0
    while n<N:
        n=n+1
        u=.........
    return u


def recherche(seuil):
    u,n=500,0
    while U(n).........:
        n=n+1
        u=.........
    return (n,u)
```

<u>Exercice 3 :</u> La suite de Fibonnaci

La suite de Fibonacci est une suite d'entiers dans laquelle chaque terme est la somme des deux termes qui le précèdent. Elle commence généralement par les termes 0 et 1 .

Elle doit son nom à **Leonardo Fibonacci** (v. 1175 à Pise - v. 1250) un mathématicien italien qui avait pour nom d'usage « Leonardo Pisano » ou « Léonard de Pise »

Fibonacci dans un problème récréatif posé dans l'ouvrage *Liber abaci* (1202), décrit la croissance d'une population de lapins :

« *Un homme met un couple de lapins dans un lieu isolé de tous les côtés par un mur. Combien de couples obtient-on en un an si chaque couple engendre tous les mois un nouveau couple à compter du troisième mois de son existence* ? »

Leonardo Fibonacci (1175-1250)

On définit donc la suite de Fibonacci (F_n) par : $\begin{cases} F_0=0;F_1=1 \\ F_{n+2}=F_{n+1}+F_n \end{cases}$

1) Calculer $F_2 = 1$ $F_3 = 2$ $F_4 = 3$ $F_5 = 5$ $F_6 = 8$

2) Ecrire une fonction **F(n)** qui renvoie le terme de rang n ce cette suite.

```
def F(n):
    u,v,n=0,1,0
    for i in range(n) :
        w=v
        v=u+v
        u=w
    return u
```

3) On note $\Phi(n) = \dfrac{F_{n+1}}{F_n}$.

Ecrire une fonction **phi(n)** qui renvoie le rapport $\dfrac{F_{n+1}}{F_n}$ avec $n \geq 1$.

Conjecturer la limite de ce rapport.

```
def phi(n) :
    f=F(n+1)/F(n)
    return f
```

La suite semble converger vers 1,618

<u>Histoire</u>
Johannes Kepler (1571-1630 avait remarqué que le taux de croissance des nombres de Fibonacci c'est à dire le rapport $\dfrac{F_{n+1}}{F_n}$ converge vers le nombre d'or

Exercice 4 : Les factorielles

On note n! (se lit « n factoriel ») le nombre $1\times2\times3\times\ldots\times n$ pour tout entier naturel n > 0.

Par convention 0! = 1 . On a donc : $\begin{cases} 0!=1 \\ n!=1\times2\times\ldots\times n\,,\ n>0 \end{cases}$

Histoire
la notation factorielle est introduite par le mathématicien Christian KRAMP (1760-1826) en 1808 dans Eléments d'arithmétique universelle

Partie A

1) Compléter : 1! = 1 2! = 2 3! = 6 4! = 24 5! = 120

2) Soit n > 0. exprimer (n+1)! en fonction de n! : (n+1)! = n! * (n+1)

3) Ecrire une fonction **fact(n)** qui renvoie n! avec n > 0

```
def fact(n) :
    F=1
    for i in range(1,n+1):
        F = F*i
    return F
```

Partie B

On considère la suite v_n définie pour tout entier naturel n par :

$$v_n = \sum_{k=0}^{k=n} \frac{1}{k!} = \frac{1}{0!}+\frac{1}{1!}+\frac{1}{2!}+\ldots+\frac{1}{n!}$$

1) Ecrire une fonction **v(n)** qui renvoie le terme de rang n de la suite (v_n).

```
def v(n) :
    v=0
    for i in range(n+1) :
        v=v+1/fact(i)
    return v
```

2) a) Compléter alors à 10^{-3} près :

v_4 = 2,708333 v_{10} = 2,7182818... v_{100} = 2,7182818...

b) Vers quel nombre (connu) semble converger (v_n) ? vers le nombre e = e^1

3) **Bonus : optimisation**

a) A l'aide d'un compteur, évaluer le nombre de multiplications effectués pour calculer v_{10}

b) Peut-on optimiser ce programme pour faire un minimum de multiplication ?