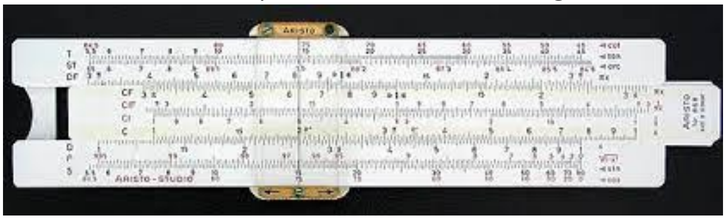
Ciel 1- Fonctions logarithmes

1- ORIGINE DES FONCTIONS LOGARITHMES

Un mathématicien écossais *John Napier* (1550 – 1617) inventa les fonctions logarithmes. Il recherchait à l'époque des fonctions mathématiques f qui permettaient d'avoir : $f(a \times b) = f(a) + f(b)$, a et b étant des nombres positifs non nuls, quelconques.

L'objectif de Napier était à l'époque de trouver une technique simple pour réaliser des multiplications complexes. Son idée était d'utiliser une telle fonction f dont l'image d'un produit est égale à la somme des images des facteurs de ce produit, afin de pouvoir transformer en quelques sortes, les produits en somme.

Concrètement cela a permis l'invention des règles à calcul :



Ses travaux ont montré que les fonctions f qui satisfaisaient $f(a \times b) = f(a) + f(b)$ étaient nombreuses. Elles avaient toutes en point commun, le fait que $f(1) = 0$ et que :

$$f(x) = \frac{1}{\int_1^k \frac{1}{x} dx} \times \int_1^x \frac{1}{x} dx$$

k étant un nombre positif quelconque.

- Si $k = e \approx 2{,}718281828$, la fonction f est appelée LOGARITHME NEPERIEN. Elle se note alors $\ln(x)$.
- Si $k = 10$, la fonction f est appelée LOGARITHME DECIMAL. Elle se note alors $\log(x)$.
- Si $k = 2$, la fonction f est appelée LOGARITHME BINAIRE. Elle se note alors $\log_2(x)$.

2- FONCTION LOGARITHME NEPERIEN LN :

La courbe ci-contre est celle de la fonction inverse $\frac{1}{x}$.

⇨ En comptant le nombre d'unité d'aires, déterminer la valeur du nombre :

$$\int_1^{2{,}172817} \frac{1}{x} dx$$

Si on fait le calcul précisément en utilisant l'algorithme d'intégration vu en cours, et donné ci-contre, on trouve :

```python
def f(x):
    return 1/x

def integrale(a,b,dx):
    x = a
    somme = 0
    while x < b :
        somme = somme + f(x)*dx
        x = x + dx
    return somme
```

```
>>> integrale(1,2.7182871828718287182871828,0.000000001)
1.0000018874622831
```

On peut donc donner le résultat suivant :

$$\int_{1}^{e} \frac{1}{x} dx = 1$$

Ainsi, la fonction logarithme népérien notée ln peut être définie par :

$$ln(x) = \frac{1}{\int_{1}^{2,17281828} \frac{1}{x} dx} \times \int_{1}^{x} \frac{1}{x} dx = \frac{1}{1} \times \int_{1}^{x} \frac{1}{x}\, dx$$

On a donc finalement :

$$ln(x) = \int_{1}^{x} \frac{1}{x}\, dx$$

On peut coder en python, une fonction notée $\ln(x)$ qui renvoie le nombre $\int_{1}^{x} \frac{1}{x}\, dx$. Le code est donné ci-contre :

```python
def f(x):
    return 1/x

def integrale(a,b,dx):
    x = a
    somme = 0
    while x < b :
        somme = somme + f(x)*dx
        x = x + dx
    return somme

def ln(x) :
    dx = 0.000001
    y = integrale(1,x,dx)
    return y
```

Si on exécute cette fonction dans la console, on obtient par exemple :

```
>>> ln(2.7182818281828)
1.000000379187599
```

Par contre, pour des valeurs de x plus petite que 1, ce code renvoie 0 car la condition de la boucle *while* dans la fonction *integrale()* est fausse dès le départ. Pour pouvoir tout de même calculer $\int_{1}^{x} \frac{1}{x}\, dx$, il est nécessaire de modifier le code de la fonction *integrale()* comme donné ci-contre.

```python
def integrale(a,b,dx):
    x = a
    somme = 0
    if b >= a :
        while x < b :
            somme = somme + f(x)*dx
            x = x + dx
    elif b < a :
        dx = - dx
        while x > b :
            somme = somme + f(x)*dx
            x = x + dx

    return somme
```

On obtient alors, par exemple pour $\ln(0.9)$:

```
>>> ln(0.9)
-0.10536046010245523
```

Il n'est cependant possible de calculer $\ln(x)$, **uniquement si $x > 0$** .

⇨ Ecrire le code des fonctions *f()* , *integrale()* et *ln()* et utiliser ces fonctions pour calculer les valeurs de :

- $\ln(1)$
- $\ln(e)$ avec $e = 2.71828182818281828$
- $\ln\left(\frac{1}{e}\right)$
- $\ln(e \times e)$
- $\ln(e \times e \times e)$
- $\ln\left(\frac{1}{e \times e}\right)$
- $\ln\left(\frac{1}{e \times e \times e}\right)$

Mettre les copies d'écran des exécutions dans votre compte-rendu. Par exemple :

```
>>> ln(1)
0
```

3- Fonction logarithme decimal log :

Comme on l'a vu précédemment, il existe plusieurs fonctions logarithmes. Celle nommée l*ogarithme décimal* ou « *logarithme de base 10* » est définie par :

$$log(x) = \frac{1}{\int_1^{10} \frac{1}{x} dx} \times \int_1^x \frac{1}{x} dx$$

Comme on sait à présent que $\int_1^x \frac{1}{x} dx = \ln(x)$, on peut simplifier cette expression de la manière suivante :

$$log(x) = \frac{1}{\ln(10)} \times \ln(x) = \frac{\ln(x)}{\ln(10)}$$

On peut ainsi coder en python, une fonction notée *log(x)* et qui renvoie le logarithme décimal de la valeur x :

```
def log(x) :
    y = ln(x) / ln(10)
    return y
```

⇨ Ecrire le code de la fonction *log()* et l'utiliser pour calculer les valeurs de :

- $\log(1)$
- $\log(10)$
- $\log\left(\frac{1}{10}\right)$
- $\log(100)$
- $\log(1000)$
- $\log\left(\frac{1}{100}\right)$
- $\log\left(\frac{1}{1000}\right)$

Mettre les copies d'écran des exécutions dans votre compte-rendu. Par exemple :

```
>>> log(10)
1.0
```

⇨ Calculer aussi les valeurs de :

- $\log(2)$
- $\log(5)$
- $\log(2) + \log(5)$

La fonction log() satisfait-elle la relation $log(a \times b) = log(a) + log(b)$ voulue par John Napier au 17[ième] siècle ? Justifier sur votre compte-rendu.

⇨ Calculer les valeurs de :

- $\log(5)$
- $\log(2)$
- $\log(5) - \log(2)$
- $\log(\frac{5}{2})$

La fonction log() satisfait-elle la relation $log\left(\frac{a}{b}\right) = log(a) - log(b)$? Justifier sur votre compte-rendu.

4- FONCTION LOGARITHME BINAIRE LOG$_2$:

La fonction *logarithme binaire* ou « *logarithme de base 2* » est définie par :

$$log_2(x) = \frac{\ln(x)}{\ln(2)}$$

On peut coder en python, une fonction notée log2(x) , qui renvoie le logarithme binaire de la valeur x :

```
def log2(x) :
    y = ln(x) / ln(2)
    return y
```

⇨ Ecrire le code de la fonction *log2()* et calculer les valeurs de :

- $log_2(1)$
- $log_2(2)$
- $log_2(4)$
- $log_2(8)$
- $log_2(16)$
- $log_2(32)$
- $log_2(256)$
- $log_2(512)$
- $log_2\left(\frac{1}{4}\right)$
- $log_2\left(\frac{1}{256}\right)$

Mettre les copies d'écran des exécutions dans votre compte-rendu. Par exemple :

```
>>> log2(2)
1.0
```

⇨ Calculer les valeurs de :

- $log_2(4)$
- $log_2(16)$
- $log_2(4 \times 16)$
- $log_2(4) + log_2(16)$

La fonction log$_2$() satisfait-elle la relation $log_2(a \times b) = log_2(a) + log_2(b)$ voulue par John Napier au 17$^{\text{ième}}$ siècle ? Justifier sur votre compte-rendu.

⇨ La fonction log$_2$() satisfait-elle la relation $log_2\left(\frac{a}{b}\right) = log_2(a) - log_2(b)$? Justifier sur votre compte-rendu.


3
2.5
2
1.5
1
0.5
0
1
2
3
4
5
6
7
8
9
10