TP Python : résolution de $f(x) = 0$

Objectif. Étant donnée une fonction $f : I \to \mathbb{R}$ continue, le problème est d'obtenir efficacement une solution de $f(x) = 0$ dans un intervalle $I = [a, b]$. La condition $f(a)f(b) < 0$ assure l'existence d'une solution (puisque dans ce cas $f(a)$ et $f(b)$ sont alors de signe opposés). Une étude théorique plus fine peut être nécessaire pour isoler une unique solution dans un intervalle.

On utilisera pour les différents algorithmes, la fonction f définie sur $\mathbb{R}$ par :

$$f(x) = -3x^3 + 6x + 3$$

On peut facilement montrer par une étude de fonction que l'équation $f(x) = 0$ admet une solution sur $[1; 2]$.

I Algorithme de Dichotomie.

Entrée : a
Entrée : b
Entrée : ϵ
Tant que $b - a > \epsilon$ faire
    $c \leftarrow \dfrac{a+b}{2}$
    Si $f(a) \times f(c) \leqslant 0$, alors
        $b \leftarrow c$
    sinon
        $a \leftarrow c$
    fin si
Fin Tant que
Afficher a et b

Exercice 1. On choisit ici $a = 1$, $b = 2$ et $\epsilon = 0,04$. Recopier et compléter le tableau ci-dessous jusqu'à arrêt de l'algorithme :

Rang	1	2	3	4	5	6
a	1	1,5				
b	2	2				
$b - a > \epsilon$	Vrai	Vrai				
c	1,5					
Signe de $f(a) \times f(c)$	+					

Exercice 2. Entrez le programme de dichotomie dans edupython :

```
def f(x) :
    return -3*x**3+6*x+3
Entrée : a = 1
Entrée : b = 2
Entrée : epsilon = 0.001
while b − a > epsilon :
    c = (a + b)/2
    if f(a) ∗ f(c) <= 0 :
        b = c
    else :
        a = c
print("a=",a," et b=",b)
```

II Exemple d'algorithme de programmation d'une suite.

Exemple 1. Pour programmer la suite (u_n) définie par $u_0 = 3$ et $u_{n+1} = g(u_n)$ avec $g(x) = \dfrac{1}{2}\left(x + \dfrac{3}{x}\right)$.

On a l'algorithme :

Définir la fonction $g(x) = \frac{1}{2}\left(x + \frac{3}{x}\right)$
Entrée : $U = 3$
Entrée : $n = 0$
Tant que $n < 10$ faire
    $U \leftarrow g(U)$
    $n \leftarrow n + 1$
    Afficher "pour" n "on a" U

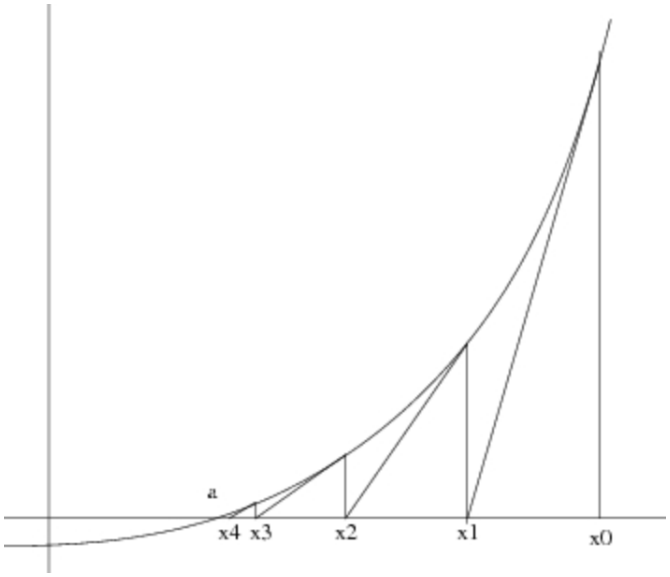
Programmer en Python :

```
def g(x) :
    return 0.5*(x+3/x)
U = 3
n = 0
while n<10 :
    U = g(U)
    n = n + 1
    print("Pour n=",n,"Un vaut",U)
```

Exercice 3. Écrire le programme précédent.

III Algorithme de Newton.

La méthode de Newton utilise les tangentes pour déterminer les solutions de $f(x) = 0$.



On définie donc une suite à partir d'une valeur initiale par exemple $x_0 = 2$. On construit la suite des points d'intersections entre la tangente au point d'abscisse x_n et l'axe des abscisses pour construire x_{n+1}.
L'équation de la tangente au point d'abscisse x_n est :

$$y = f'(x_n)(x - x_n) + f(x_n)$$

Donc pour $y = 0$, on obtient $x = x_{n+1}$:

$$0 = f'(x_n)(x_{n+1} - x_n) + f(x_n) \Leftrightarrow x_{n+1} - x_n = \frac{-f(x_n)}{f'(x_n)} \Leftrightarrow x_{n+1} = x_n - \frac{f(x_n)}{f'(x_n)}$$

Exercice 4. Écrire l'algorithme de programmation de la suite précédente, puis programmer l'algorithme pour obtenir les 10 premiers termes de la suite (x_n).

Exercice 5. Utiliser les deux méthodes d'approximation pour obtenir une solution des équations suivantes sur l'intervalle donné :

1. $(E_1) : -3x^5 - 3x^3 + 9x^2 + 3x - 3 = 0$ sur $[-1, 0]$, puis sur $[0, 1]$, puis enfin $[1, 2]$.
2. $(E_2) : \sin(x) = 0$ sur $[3, 4]$. (Il faudra utiliser un package Python : "from math import sin" en entrée du programme)