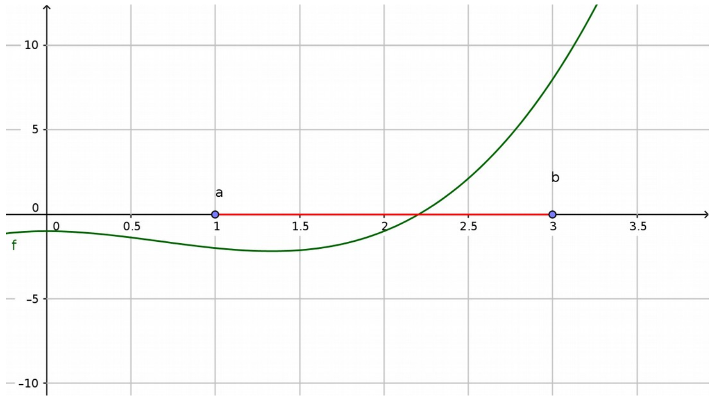
ACTIVITÉ SUR LA COMPLEXITÉ

On veut trouver un « zéro » d'une fonction grâce à un algorithme par dichotomie et aussi par un autre à balayage.
On suppose que la fonction ne s'annule qu'une seule fois sur [a,b] et que *f(a)* et *f(b)* sont de signes différents.
Exemple :



Partie A : Balayage
1) « A la main », en progressant de 1 vers 3 avec de intervalles d'une amplitude de 0,5, dans quel intervalle d'amplitude 0,5 se trouve x_0 tel que $f(x_0)=0$? De la même façon, estimer un intervalle d'amplitude 0,1 qui contient x_0.
2) En s'inspirant de la méthode précédente, écrire une fonction en Python qui prend comme argument, l'amplitude voulue, les valeurs a et b de l'intervalle [a;b] et éventuellement la fonction et qui renvoie un intervalle ayant l'amplitude voulue et qui contient la solution
3) Tester la fonction avec $f(x)=x^3-2x^2-1$ sur l'intervalle [1;3] avec différentes amplitudes comme 10^{-1}, 10^{-2} etc.
4) Où placer un compteur pour savoir combien de fois la boucle est exécutée ?
5) Compter le nombre total d'opérations élémentaires (assignations, opérations arithmétiques, calculs de booléens) effectuées pour 10^{-1}, $10^{-2,}10^{-3}$ et ... 10^{-n} .

Partie B : Dichotomie
1) « A la main », dans quelle moitié d'intervalle se trouve la solution ? Dans quelle moitié de l'intervalle précédent se trouve la solution ? Quelle est son amplitude ?
2) En s'inspirant de la méthode précédente, écrire une fonction en Python qui prend comme argument, l'amplitude voulue, les valeurs a et b de l'intervalle [a;b] et éventuellement la fonction et qui renvoie un intervalle ayant l'amplitude voulue et qui contient la solution
3) Tester la fonction avec $f(x)=x^3-2x^2-1$ sur l'intervalle [1;3] avec différentes amplitudes comme 10^{-1}, 10^{-2} etc.
4) Où placer un compteur pour savoir combien de fois la boucle est exécutée ?
5) Quelle est l'amplitude de l'intervalle pour 10 itérations ?
6) Compter le nombre total d' opérations élémentaires (assignations, opérations arithmétiques, calculs de booléens) effectuées pour 10^{-1}, $10^{-2,}10^{-3}$ et ... 10^{-8} .

Bilan :
Comparer pour chaque algorithme le nombre d' opérations élémentaires pour 10^{-3} et 10^{-8} .
Pour quel algorithme cela augmente le plus vite quand on fait augmenter n pour une précision à 10^{-n} ?
Ces valeurs représentent le temps d'exécution de ces algorithmes.
On dira que le premier algorithme a une plus grande **complexité** que le second.

Exemple d’attendu de la part des élèves :

```python
import math
def dichotomie(f,e,a,b):
    n=0
    while abs(b-a)>e:
        milieu=(a+b)/2
        n+=1
        if f(a)*f(milieu)>0:
            a=milieu
        else :
            b=milieu
    return (a,b),n

def balayage(f,e,a,b):
    n=0
    while(a<=b):
        n+=1
        if f(a)*f(b)>0:
            return (a-e,a),n
        a+=e

#dichotomie(lambda x:x*math.exp(x)-1,0.1,0,1)
#dichotomie(lambda x:x**3-2*x**2-1,0.001,1,3)

```