PYTHON : TROUVER UN ZÉRO D'UNE FONCTION CONTINUE

EXERCICE 1 (Par dichotomie)

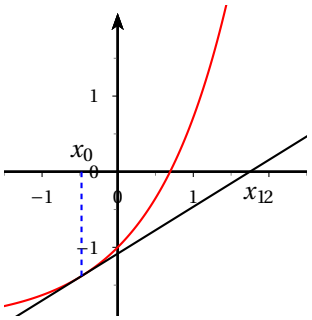
1) Programmer la fonction **dichotomie**(`f,a,b,eps=1e-3`) qui prend en arguments `f,a,b` et l'argument optionnel `eps`[1] et qui recherche la valeur approchée d'un zéro de `f` par dichotomie entre `a` et `b` avec une erreur maximale de `eps` (c'est-à-dire |b-a|≤eps).

2) Tester la fonction dichotomie avec la fonction $x \mapsto x^2 - x - 1$ entre 1 et 2.
Vérifier que les résultats correspondent à ceux calculés analytiquement.

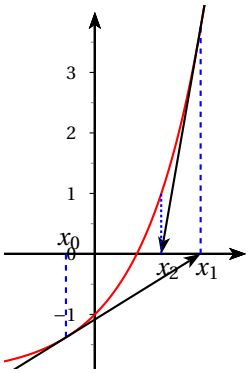
Approfondissement facultatif :
Pour définir une fonction directement en argument de `dichotomie`, on peut utiliser la syntaxe « `lambda x:`*`formule`* ».
Ainsi, on écrit : `dichotomie(lambda x:x**2-x-1,1,2)`.

EXERCICE 2 (Par la méthode de Newton)

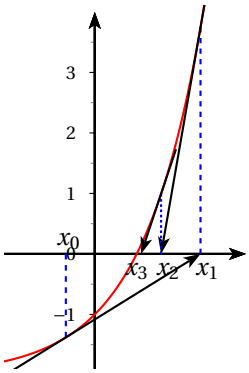
Etape 1 : On choisit un point x_0 et on trace la tangente à la courbe au point $(x_0, f(x_0))$.
Cette tangente coupe l'axe des abscisses en x_1.



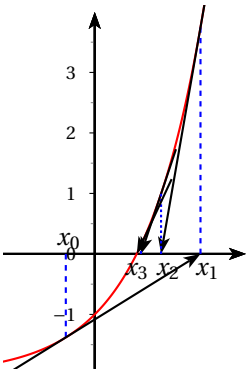
Pour les étapes suivantes, on réitère la méthode jusqu'à être suffisamment proche du point où f s'annule.



Étape 2



Étape 3



Étape 4...

1) Donner l'expression de x_{n+1} en fonction de x_n.

2) Programmer la fonction Python **newton**(`f,fp,a,n`) qui calcule la valeur de x_n à partir de x_0 =`a` avec la fonction `f` de dérivée `fp`.

3) Tester l'algorithme précédent avec $x \mapsto x^2 - x - 1$.

4) Tester l'algorithme précédent avec $x \mapsto x^2 - 2$ pour trouver une valeur approchée de $\sqrt{2}$.

5) Modifier la fonction Python pour qu'elle prenne un argument optionnel `eps` (qui vaut 10^{-3} par défaut) à la place de n de telle sorte qu'elle s'arrête lorsque $|x_n - x_{n-1}| \leq$ `eps`.

6) Modifier la fonction Python pour qu'elle trace la fonction et les différents traits de construction (comme dans les graphiques plus haut).

EXERCICE 3 (Par la méthode de la sécante)
La méthode de Newton nécessite d'avoir accès à la dérivée. La méthode de la sécante permet de s'en affranchir en approximant la dérivée par la pente de la corde entre x_n et x_{n-1} pour trouver x_{n+1}.

[1] La syntaxe indique à Python que `eps` vaut 10^{-3} si aucune valeur n'est donnée en argument

1) Donner l'expression de x_{n+1} en fonction de x_n et de x_{n-1}.

2) Programmer la fonction Python **secante(**`f,a,b,n`**)** qui calcule la valeur de x_n à partir de $x_0 = a$ et $x_1 = b$ avec la fonction `f`.

3) Tester l'algorithme précédent avec $x \mapsto x^2 - x - 1$.

4) Tester l'algorithme précédent avec $x \mapsto x^2 - 2$ pour trouver une valeur approchée de $\sqrt{2}$.

5) Modifier la fonction Python pour qu'elle prenne un argument optionnel `eps` (qui vaut 10^{-3} par défaut) à la place de n de telle sorte qu'elle s'arrête lorsque $|x_n - x_{n-1}| \leq$ `eps`.

EXERCICE 4 (Application à la recherche dans une liste triée)

1) Programmer un algorithme **recherche(**`liste,a`**)** qui recherche la valeur « `a` » dans la liste « `liste` » et qui renvoie son indice.
La fonction renvoie `len(liste)` si `a` n'est pas dans `liste`.

2) On suppose que la liste est triée, programmer un algorithme **dichotomieListe(**`liste,a`**)** qui recherche par dichotomie l'indice de la valeur a dans la liste triée « `liste` ». La fonction renvoie une erreur si la valeur n'est pas dans la liste.

3) Essayer avec la recherche dans $[0,1,2,3,4,5,6]$ de 0, de 1, de 1,5 et de 2.

4) Générer la liste (ordonnée) des nombres de 1 à 99 999, puis comparer les temps de recherche entre l'algorithme *normal*, et celui par dichotomie. Faire différents essais en fonction de la position dans la liste.
On pourra utiliser la fonction `clock()` de la bibliothèque `time`.

EXERCICE 5 (Pour ceux qui ont terminé)
Trouver la valeur de $\sqrt{2}$ en utilisant une suite récurrente.
On pourra s'aider de la fonction $x \mapsto -\frac{x^2}{2} + 1 + x$ dont on cherchera un point fixe.