académie Grenoble
RÉGION ACADÉMIQUE AUVERGNE-RHÔNE-ALPES
MINISTÈRE DE L'ÉDUCATION NATIONALE ET DE LA JEUNESSE
MINISTÈRE DE L'ENSEIGNEMENT SUPÉRIEUR, DE LA RECHERCHE ET DE L'INNOVATION

Approximation d'un nombre par une suite

Ce document a été élaboré par les formateurs du groupe académique lycée. Il est destiné aux enseignants et n'a pas vocation à être distribué en l'état aux élèves.

Rappel concernant la notion de convergence d'ordre p.

Il existe plusieurs définitions qui permettent d'appréhender la vitesse de convergence d'une suite vers sa limite. Nous proposons ici la notion d'ordre de convergence.

Définition : Soit (x_n) une suite convergeant vers un réel a, avec $x_n \neq a$ pour tout entier n. On dit que la convergence de la suite (x_n) vers a est d'ordre r (r entier naturel non nul) si le rapport

$\frac{|x_{n+1}-a|}{|x_n-a|^r}$ tend vers k, où $0 < k < 1$.

Remarque : lors d'une convergence dite quadratique ($r = 2$), on remarque que, à partir d'un certain rang, si l'erreur commise au rang n (c'est-à-dire $|x_n - a|$) est inférieure à 10^{-2} alors l'erreur commise au rang suivant ($|x_{n+1} - a|$) sera inférieure à 10^{-4}.

Présentation de plusieurs algorithmes présents dans les programmes de spécialité de mathématiques ou mathématiques complémentaires et propositions d'activités à adapter selon les classes

Méthode de Héron : Héron d'Alexandrie est un ingénieur, un mécanicien et un mathématicien grec du 1er s. après J.-C. Cette méthode permet l'approximation de $\sqrt{a}$ avec une convergence quadratique.

Par exemple, pour $\sqrt{2}$, en partant de $a = 1$, on obtient en deux itérations une valeur approchée à 10^{-2} près. Avec trois itérations, on obtient une valeur approchée à 10^{-5} près et avec quatre itérations une valeur approchée à 10^{-11} près.

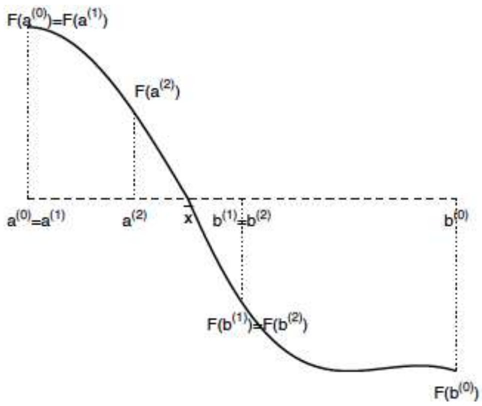
Pour aller plus loin : https://maths.discip.ac-caen.fr/spip.php?article466

http://irem.univ-reunion.fr/spip.php?article919

Aller à l'activité

Méthode de dichotomie : cette méthode pour approcher une solution de l'équation $f(x) = 0$ est simple à mettre en œuvre. En terminale, on donne souvent le TVI en premier que l'on utilise pour justifier la convergence de l'algorithme de dichotomie. Mais dans le supérieur on peut utiliser l'algorithme de dichotomie pour démontrer le TVI.

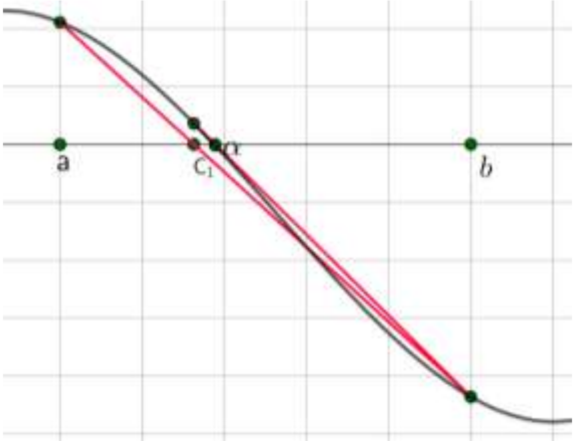
La méthode de dichotomie est simple à mettre en œuvre mais la convergence n'est que d'ordre 1.



Aller à l'activité

Prolongement : la méthode de la fausse position a été souvent utilisée en Occident à partir du 12ème siècle, notamment par Fibonacci. L'algorithme converge sous de bonnes hypothèses (par exemple la convexité). Elle n'est pas explicitement au programme. D'un point de vue pédagogique, elle peut être intéressante pour prolonger la méthode de dichotomie et pour introduire la méthode de la sécante.

On procède de la même façon que pour la méthode par dichotomie mais au lieu de prendre le centre de l'intervalle, on prend à chaque étape l'intersection entre l'axe des abscisses et la sécante passant par les points qui ont pour abscisses les bornes de l'intervalle. La méthode de la fausse position est un peu plus performante que celle de la dichotomie mais moins que celles qui suivent. Elle est super-linéaire, c'est-à-dire que si on note α la solution exacte, la suite $\left(\frac{x_{n+1}-\alpha}{x_n-\alpha}\right)$ tend



vers 0. L'inconvénient, par rapport à la dichotomie, est que l'intervalle dans lequel se situe le zéro n'a pas forcément une amplitude qui converge vers 0. (C'est le cas notamment lorsque la fonction est (localement) convexe ou concave). Dans ce cas, il y a une des deux bornes qui ne change pas. On peut améliorer cela : notons a et b les bornes de l'intervalle, si c'est la borne b qui ne change pas, au lieu de prendre l'intersection entre l'axe des abscisses et la droite passant par $A\big(a, f(a)\big)$ et $B\big(b, f(b)\big)$ on peut prendre l'intersection entre l'axe des abscisses et la droite passant par $A\big(a, f(a)\big)$ et $B'\left(b, \frac{f(b)}{2}\right)$.

Pour aller plus loin :
https://www.ceremade.dauphine.fr/~legendre/enseignement/methnum/notes_cours_20200327.pdf

https://www.math.univ-paris13.fr/~cuvelier/docs/Enseignements/MACS1/AnaNumI/15-16/AnaNum_macs1_12-10-2015.pdf (page 53)

Aller à l'activité

Méthode de la sécante : cette méthode ne nécessite pas plus de conditions que la méthode de dichotomie. On ne doit pas la confondre avec la méthode de la fausse position même si elles se ressemblent. Ici, on approche une solution de l'équation $f(x) = 0$ en calculant x_{n+2} comme l'abscisse du point d'intersection de l'axe des abscisses et de la droite passant par les points $(x_n, f(x_n))$ et $(x_{n+1}, f(x_{n+1}))$.

Par rapport à la méthode précédente, la solution n'est pas forcément dans l'intervalle de bornes x_{n+1} et x_n.

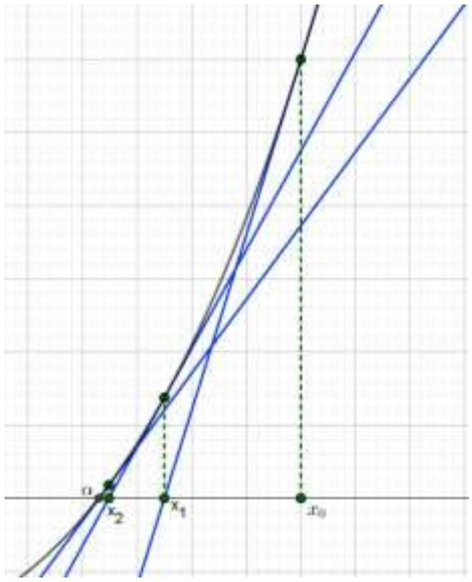
La convergence est d'ordre $\varphi = \frac{1+\sqrt{5}}{2} \approx 1{,}62$. (Sous l'hypothèse que la fonction soit C^2).

Pour aller plus loin : http://www.unige.ch/~alaurent/Agregation/Liste_DVP/Methode_secante.pdf

https://www.math.univ-paris13.fr/~cuvelier/docs/Enseignements/MACS1/AnaNumI/15-16/AnaNum_macs1_12-10-2015.pdf (page 52)

Aller à l'activité

Méthode de Newton (ou méthode de Newton-Raphson) :



L'avantage majeur de la méthode de Newton par rapport aux autres méthodes proposées est sa convergence (locale) d'ordre 2 (quadratique). L'inconvénient majeur tient au fait que la fonction doit être dérivable, avec une dérivée non nulle et pouvant être facilement calculée. Par ailleurs, si la valeur de départ est « trop éloignée » de la solution cherchée, la méthode de Newton peut ne pas converger. Ainsi, plus encore que pour tout autre méthode, une mise en œuvre correcte de la méthode de Newton doit-elle inclure un code de contrôle du nombre d'itérations.

Ici, on approche une solution de l'équation $f(x) = 0$ en calculant x_{n+1} comme l'abscisse du point d'intersection de l'axe des abscisses et de la tangente à la courbe représentative de f en x_n.

Remarque : Soit a un réel positif. Avec $f(x) = x^2 - a$, on retrouve la méthode de Héron. En effet, pour la méthode de Newton, on considère la suite $x_{n+1} = x_n - \frac{f(x_n)}{f'(x_n)}$ qui dans ce cas particulier donne $x_{n+1} = x_n - \frac{{x_n}^2 - a}{2x_n} = \frac{{x_n}^2 + a}{2x_n} = \frac{x_n + \frac{a}{x_n}}{2}$, ce qui correspond bien à la suite définie dans la méthode de Heron.

Pour aller plus loin : https://www.apmep.fr/IMG/pdf/APMEP_article_BV_7_C_A.pdf (Très intéressant tant du point de vue de la méthode elle-même que de son approche historique.)

Aller à l'activité

Algorithme de Brouncker : mathématicien et linguiste anglais, William Brouncker (1620-1684) fut avec John Wallis un des fondateurs de la Royal Society, l'équivalent de notre « Académie des Sciences ». Il en fut même le président.

Cet algorithme n'a plus beaucoup d'intérêt à notre époque mais en 1668, il permet de donner le premier développement en série d'une valeur de ln (ici ln(2)). À partir de cette idée, John Wallis généralise le développement en série de $\ln(1 + x)$.

Le programme de mathématiques complémentaires évoque une « approximation de $ln(2)$ par dichotomie selon l'algorithme de Brouncker ». Cela fait référence au découpage de l'aire sous l'hyperbole en rectangles venant « boucher les trous » par dichotomie. Voir « https://www.geogebra.org/m/dcbuqkjx ».

Aller à l'activité

Algorithme de Briggs : mathématicien anglais, Briggs (1561-1630) travailla avec John Napier à l'invention des logarithmes. L'algorithme de Briggs a permis de créer les tables du logarithme. La méthode utilisée par Briggs est à la base de l'algorithme CORDIC utilisé aujourd'hui dans les calculatrices. (Source : http://publimath.irem.univ-mrs.fr/glossaire/TA013.htm).

Aller à l'activité

La méthode de Héron

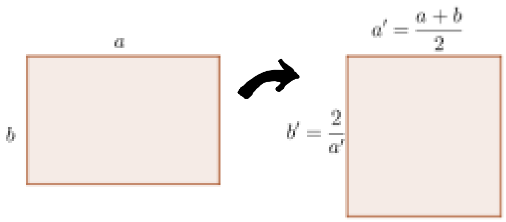
(mathématicien grec du I^er siècle avant J.C.)

Cas particulier : calcul d'une valeur approchée de $\sqrt{2}$

Partie A : Principe géométrique

Géométriquement $\sqrt{2}$ est la mesure du côté d'un carré dont l'aire est 2. Pour obtenir une valeur approchée de $\sqrt{2}$, la méthode de Héron consiste à partir d'un rectangle dont l'aire est 2 (Par exemple le rectangle de côté 1 par 2). On lui applique un certain nombre de fois la transformation décrite ci-dessous, afin de le rapprocher peu à peu du carré de côté $\sqrt{2}$.

a et b sont les longueurs des côtés à l'étape n et a' et b' à l'étape suivante.



Justifier que cette transformation conserve l'aire.

On suppose $b < a$. Montrer que $a' < a$ et que $b' > b$.

Combien d'étapes faut-il pour obtenir une valeur approchée de $\sqrt{2}$ à 10^{-2} près ?

Partie B : On considère la suite (a_n) définie par $a_0 = 2$ et pour tout entier naturel n, $a_{n+1} = \frac{a_n + \frac{2}{a_n}}{2}$, ainsi que la suite (b_n) définie pour tout entier naturel n par $b_n = \frac{2}{a_n}$.

On note R_n le rectangle de côtés a_n et b_n.

1. Justifier que, pour tout entier naturel n, l'aire de R_n vaut 2.
2. Montrer que pour tout entier naturel n non nul, $a_n - \sqrt{2} = \frac{(a_{n-1} - \sqrt{2})^2}{2a_{n-1}}$.
3. En déduire que pour tout entier naturel n non nul, $a_n \geq \sqrt{2}$.
4. Montrer que (a_n) est décroissante.
5. Prouver que (a_n) est convergente puis déterminer sa limite.
6. Montrer que (b_n) est convergente puis déterminer sa limite. Que peut-on en déduire géométriquement ?
7. Donner une valeur approchée de $\sqrt{2}$ à 10^{-20} près.
8. Prolongement : on peut montrer que, pour tout entier naturel n, $a_{n+1} - \sqrt{2} \leq (a_n - \sqrt{2})^2$. Ainsi en déduire que si a_n donne une valeur approchée de $\sqrt{2}$ à 10^{-p}, alors a_{n+1} donne une valeur approchée à au moins 10^{-2p}.

On peut aussi profiter de cet exercice pour introduire la notion de suites adjacentes.

Pour approfondir : https://maths.discip.ac-caen.fr/spip.php?article466

[Retour présentation algorithmes]()

La méthode de dichotomie

Calcul d'une valeur approchée d'une solution d'équation, notamment pour obtenir la valeur approchée d'un irrationnel.

Partie A :

1. Écrire une fonction Python qui demande deux nombres et qui renvoie True s'ils sont de même signe et False sinon. Nommez cette fonction meme_signe.
2. On définit, en Python, la fonction f de la façon suivante :

```
def f(x):
    return x**3-2
```

   a) Que donne dans la console meme_signe(f(0),f(1)) ?
   b) Que donne dans la console meme_signe(f(1),f(2)) ?

Partie B : On considère la fonction f définie par $f(x) = x^3 - 2$ sur $\mathbb{R}$. On cherche à trouver une valeur approchée

1. Justifier que l'équation $f(x) = 0$ admet une unique solution α sur $\mathbb{R}$. Justifier que celle-ci appartient à l'intervalle $[1; 2]$.
2. On découpe l'intervalle en deux intervalles de même amplitude. On obtient donc les intervalles $[1; 1{,}5]$ et $[1{,}5; 2]$. Dans lequel de ces deux intervalles se situe α ?
3. Le principe de la dichotomie est le suivant. On redécoupe l'intervalle trouvé précédemment en deux intervalles de même amplitude et on garde l'intervalle où se situe α. On réitère ce procédé jusqu'à ce que l'amplitude de l'intervalle retenu soit plus petite que 10^{-p} où p est un entier naturel fixé.
   Ecrire en Python une fonction qui prend comme paramètre p et qui renvoie un encadrement de α d'amplitude 10^{-p}.
4. Programmer cet algorithme et donner une valeur approchée à 10^{-3} près de α.
   (Remarque : La valeur de α est la racine cubique de 2 et se note $\sqrt[3]{2}$)

Partie C : On considère une fonction f continue et strictement monotone sur un intervalle $[a; b]$ et qui change de signe sur $[a; b]$. On sait donc que l'équation $f(x) = 0$ a une unique solution sur $[a; b]$.

1. Écrire en Python un algorithme qui demande comme paramètres a, b et p et qui renvoie un encadrement à 10^{-p} de la solution.
2. Faire fonctionner cet algorithme pour donner une valeur approchée à 10^{-10} près de $\sqrt{2}$.

Retour présentation algorithmes

La méthode de la fausse position

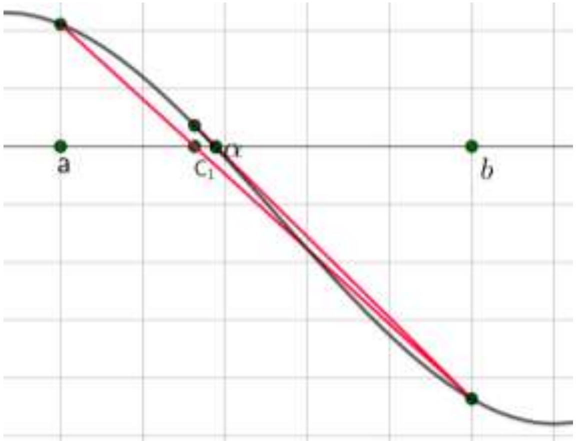
Calcul d'une valeur approchée d'une solution d'équation, notamment pour obtenir la valeur approchée d'un irrationnel.

Partie A : (Préliminaire)

1. Soit $A(1,-2)$ et $B(5,3)$. Déterminez l'abscisse du point d'intersection entre l'axe des abscisses et la droite (AB).
2. Soit la fonction f définie par $f(x) = x^2 - 4x - 2$. Déterminer l'abscisse du point d'intersection entre l'axe des abscisses et la droite (AB) où $A(2, f(2))$ et $B(5, f(5))$.
3. Soit f une fonction continue et strictement monotone sur un intervalle $[a; b]$.
   Ecrire une fonction Python qui a pour arguments deux nombres a et b et qui renvoie l'abscisse du point d'intersection entre l'axe des abscisses et la droite passant par $A(a, f(a))$ et $B(b, f(b))$.

Partie B : On considère une fonction f continue et strictement monotone sur un intervalle $[a; b]$ et qui change de signe sur $[a; b]$.

1. Expliquez brièvement pourquoi l'équation $f(x) = 0$ admet une unique solution sur $[a; b]$. On notera cette solution α
2. *Le principe de la fausse position est le suivant :*
   On note c_1 l'abscisse du point d'intersection entre l'axe des abscisses et la droite passant par $A(a, f(a))$ et $B(b, f(b))$.
   On découpe l'intervalle $[a; b]$ en deux intervalles : $[a; c_1]$ et $[c_1; b]$. Comment savoir simplement lequel des deux contient α ?
   On garde l'intervalle dans lequel se situe α.
   On réitère successivement ce procédé en gardant à chaque fois l'intervalle qui contient α.



   On note c_2, c_3,…,c_n les abscisses successives des points d'intersection entre l'axe des abscisses et les droites passant par les deux points de la courbe de f d'abscisse les bornes de l'intervalle.
   Critère d'arrêt : On pourra s'arrêter après un certain nombre d'itération, mais dans ce cas on ne connaîtra pas la précision de notre valeur approchée.
   On peut aussi s'arrêter lorsque $f(c_n - 10^{-p})$ et $f(c_n + 10^{-p})$, où p est un entier naturel fixé, sont de signes contraires. Expliquez cette condition d'arrêt.

   Ecrire en Python une fonction d'argument p qui renvoie une valeur approchée de α à 10^{-p} près.

Partie C :

1. Faire fonctionner l'algorithme précédent pour donner une valeur approchée à 10^{-10} près de $\sqrt{2}$.
2. On reprend la fonction f définie par $f(x) = x^3 - 2$ sur $\mathbb{R}$ de l'activité « la méthode de la dichotomie ».
   A l'aide de l'algorithme de la fausse position, donner une valeur approchée à 10^{-10} près de $\sqrt[3]{2}$.

Retour présentation algorithmes

La méthode de la sécante

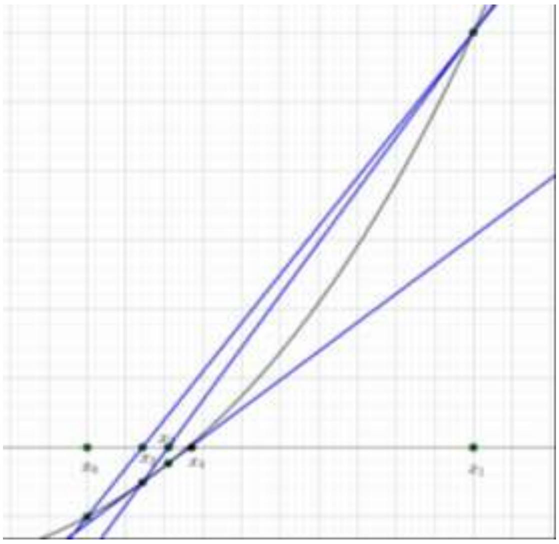
Calcul d'une valeur approchée d'une solution d'équation, notamment pour obtenir la valeur approchée d'un irrationnel.

Partie A : (Préliminaire)

1. Soit $A(1,-2)$ et $B(5,3)$. Déterminer l'abscisse du point d'intersection entre l'axe des abscisses et la droite (AB).
2. Soit la fonction f définie par $f(x) = x^2 - 3x - 2$. Déterminer l'abscisse du point d'intersection entre l'axe des abscisses et la droite (AB) où $A\big(1, f(1)\big)$ et $B\big(4, f(4)\big)$.
3. Soit f une fonction continue strictement monotone sur un intervalle $[a; b]$. Écrire une fonction Python d'arguments a et b qui renvoie l'abscisse du point d'intersection entre l'axe des abscisses et la droite passant par $A\big(a, f(a)\big)$ et $B\big(b, f(b)\big)$.

Partie B : On considère une fonction $f$continue strictement monotone sur un intervalle $[a; b]$ et qui change de signe sur $[a; b]$.

D'après le théorème des valeurs intermédiaires, on sait que l'équation $f(x) = 0$ admet une unique solution α sur l'intervalle $[a; b]$.



1. Le principe de la sécante est le suivant : On pose $x_0 = a$ et $x_1 = b$ et pour tout entier naturel n, x_{n+2} est l'abscisse du point d'intersection entre l'axe des abscisses et la droite passant par $A_n\big(x_n, f(x_n)\big)$ et $A_{n+1}\big(x_{n+1}, f(x_{n+1})\big)$.
   a) Déterminer x_2.
   b) Prouver que pour tout entier naturel n,
   $x_{n+2} = x_n - \frac{x_{n+1} - x_n}{f(x_{n+1}) - f(x_n)} f(x_n)$.
   On peut démontrer (difficilement) que (x_n) converge vers la solution α.
   Par rapport à la méthode de la fausse position, on n'a pas d'intervalle dans lequel se situe α.
   Critère d'arrêt : On pourra s'arrêter après un certain nombre d'itération, mais dans ce cas on ne connaîtra pas la précision de notre valeur approchée.
   On peut aussi s'arrêter lorsque $f(x_n - 10^{-p})$ et $f(x_n + 10^{-p})$, où p est un entier naturel fixé, sont de signes contraires. (Expliquez cette condition d'arrêt).

2. Écrire un algorithme utilisant la méthode de la sécante d'arguments a, b et p qui renvoie une valeur approchée à 10^{-p} de α.

Partie C :

1. Faire fonctionner l'algorithme précédent pour donner une valeur approchée à 10^{-10} près de $\sqrt{2}$.
2. À l'aide de l'algorithme de la fausse position, donner une valeur approchée à 10^{-10} près de $\sqrt[3]{2}$.

Retour présentation algorithmes

La méthode de Newton

Calcul d'une valeur approchée d'une solution d'équation, notamment pour obtenir la valeur approchée d'un irrationnel.

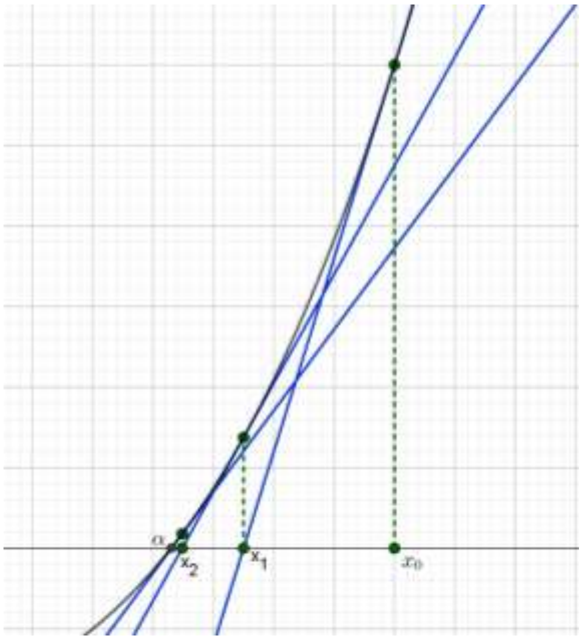
Partie A : (Préliminaires)

1. a) Soit la fonction f définie par $f(x) = x^2 - 3x - 2$. Déterminer une équation de la tangente à la courbe de f au point d'abscisse 4.
   b) Déterminer les coordonnées du point d'intersection de cette tangente avec l'axe des abscisses.

2. Soit f une fonction dérivable et strictement monotone sur un intervalle $[a\,;b]$. Ecrire une fonction Python d'argument a qui renvoie l'abscisse du point d'intersection entre l'axe des abscisses et la tangente à la courbe de f au point d'abscisse a.

Partie B : On considère une fonction f continue, dérivable, dont la dérivée ne s'annule pas sur un intervalle $[a; b]$ et qui change de signe sur $[a; b]$.

1. Expliquez brièvement pourquoi l'équation $f(x) = 0$ admet une unique solution sur $[a; b]$, on notera cette solution α.



2. Le principe de Newton est le suivant :
   On part de l'une des deux extrémités, par exemple $x_0 = b$. On pose x_1 l'abscisse du point d'intersection de la tangente à la courbe de f au point x_0 avec l'axe des abscisses. Et on réitère le procédé.

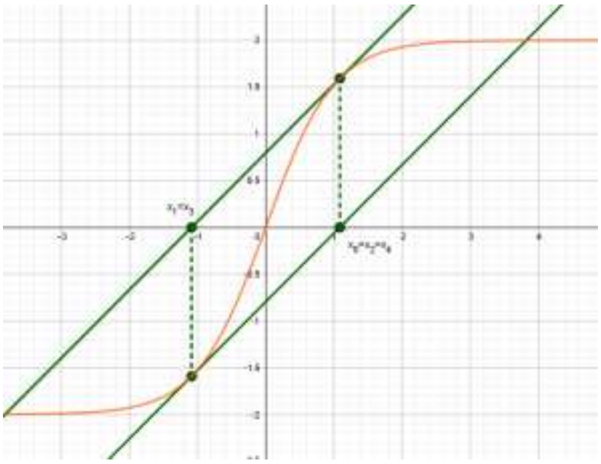
   Prouver que pour tout entier naturel n, $x_{n+1} = x_n - \frac{f(x_n)}{f'(x_n)}$.

3. Critère d'arrêt : On pourra s'arrêter après un certain nombre d'itérations, mais dans ce cas on ne connaîtra pas la précision de la valeur approchée cherchée. On peut aussi s'arrêter lorsque $f(x_n - 10^{-p})$ et $f(x_n + 10^{-p})$, où p est un entier naturel fixé, sont de signes contraires. (Expliquez cette condition d'arrêt).

Partie C :

1. Faire fonctionner l'algorithme précédent pour donner une valeur approchée à 10^{-10} près de $\sqrt{2}$.
2. On reprend la fonction f définie par $f(x) = x^3 - 2$ sur $\mathbb{R}$ de l'activité « la méthode de la dichotomie ».
   À l'aide de l'algorithme de Newton, donner une valeur approchée à 10^{-10} près de $\sqrt[3]{2}$.

Complément :



- La méthode peut diverger. (Voir figure ci-contre où l'on rentre dans une boucle infinie).
- Sous de « bonnes » hypothèses, on est certain de la convergence de la suite (x_n) et celle-ci est d'ordre 2. (c'est-à-dire que la précision de l'approximation double à chaque étape)

[Retour présentation algorithmes]

L'algorithme de Brouncker

On note $A_n = 1 - \frac{1}{2} + \frac{1}{3} - \cdots - \frac{(-1)^{n+1}}{n} = \sum_{k=1}^{n} \frac{(-1)^{k+1}}{k}$

Première version : Utilisation de la série Harmonique

$$H_n = 1 + \frac{1}{2} + \frac{1}{3} + \cdots + \frac{1}{n} = \sum_{k=1}^{n} \frac{1}{k}$$

1. a) Montrer que $A_{2n} = H_{2n} - H_n$.

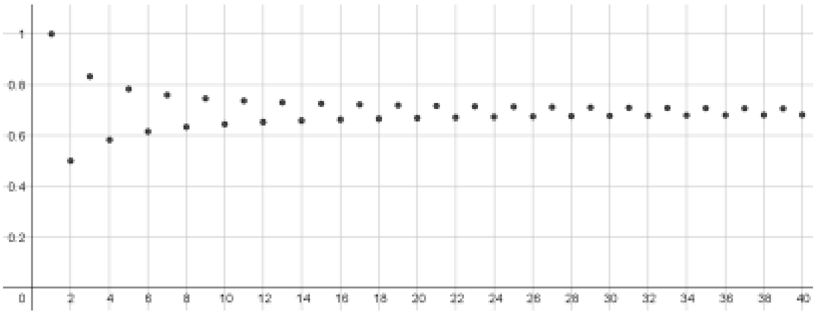
   On admet que $\lim_{n \to +\infty} (H_n - \ln(n)) = \gamma$ (où γ est un réel appelé « constante d'Euler-Mascheroni »).
   b) Montrer que $A_{2n} = H_{2n} - \ln(2n) - (H_n - \ln(n)) + \ln(2)$. En déduire la limite de (A_{2n}) .
2. Soit n un entier naturel non nul. Montrer que $A_{2n} = \frac{1}{2\times 1} + \frac{1}{3\times 4} + \cdots + \frac{1}{(2n+1)(2n+2)}$
3. Compléter les fonctions ci-dessous écrites en Python. La première correspond à l'algorithme de Brouncker qui donne une valeur approchée de $\ln(2)$. La deuxième donne le nombre d'étapes nécessaires afin d'avoir une valeur approchée à 10^{-p} de ln(2) (qui se note log(2) en Python dans la bibliothèque math)

```
def brouncker(n):
    S = 0
    for k in range(n+1):
        S =
    return
```

```
def etape_brouncker(p):
    n =
    S =
    while abs(S-log(2)) >           :
        S =
        n =
    return n,S
```

Deuxième version : Utilisation de suites adjacentes et de la série Harmonique

$$H_n = 1 + \frac{1}{2} + \frac{1}{3} + \cdots + \frac{1}{n} = \sum_{k=1}^{n} \frac{1}{k}$$



On note pour tout entier n non nul, $u_n = A_{2n}$ et $v_n = A_{2n+1}$.

1. a) Montrer que (u_n) est croissante et (v_n) décroissante.
   b) Déterminer $\lim_{n \to +\infty} v_n - u_n$. Que peut-on en déduire ?
2. Montrer que $u_n = H_{2n} - H_n$.
   On admet que $\lim_{n \to +\infty} (H_n - \ln(n)) = \gamma$ (où γ est un réel appelé « constante d'Euler-Mascheroni »).

3. Montrer que $u_n = H_{2n} - \ln(2n) - (H_n - \ln(n)) + \ln(2)$.
   En déduire la limite de (u_n) et de (v_n).
   On admettra que cela implique que (A_{2n}) est convergente et a la même limite que (u_n).
4. Soit n un entier naturel non nul. Montrer que $u_n = \frac{1}{2\times 1} + \frac{1}{3\times 4} + \cdots + \frac{1}{(2n+1)(2n+2)}$
5. Compléter la fonction ci-contre écrite en Python afin qu'elle renvoie une valeur approchée de $\ln(2)$.

```
def brouncker(n):
    S = 0
    for k in range(n+1):
        S =
    return
```

Troisième version : Approfondissement dans le cadre d'une différenciation

Soit n un entier naturel non nul. On pose pour $x \in]-1;1]$,

$$A_n(x) = x - \frac{x^2}{2} + \frac{x^3}{3} + \cdots + \frac{(-1)^{n-1}}{n} x^n = \sum_{k=1}^{n} \frac{(-1)^{k-1}}{k} x^k$$

On admet que $\forall x \in]-1;1]$, $\lim\limits_{n\to+\infty} A_n(x)$ existe et on pose $A(x) = \lim\limits_{n\to+\infty} A_n(x)$. On définit ainsi sur $]-1;1]$ une fonction notée A . On admet que A est continue sur $]-1\,;\,1]$ et dérivable sur $]-1;1[$ et que sa dérivée vérifie : $\forall x \in]-1;1[$, $A'(x) = \lim\limits_{n\to+\infty} A'_n(x)$.

1. Montrer que pour $x \in]-1;1[$, $A'_n(x) = \frac{1-(-x)^n}{1+x}$. En déduire $A'(x)$.
2. En déduire $A(x)$ pour $x \in]-1;1]$.
3. En déduire la valeur de $A(1)$.
4. Soit n un entier naturel non nul. Montrer que $A_{2n}(1) = \frac{1}{2\times 1} + \frac{1}{3\times 4} + \cdots + \frac{1}{(2n+1)(2n+2)}$
5. Compléter la fonction écrite ci-dessous en Python afin qu'elle renvoie une valeur approchée de $\ln(2)$.

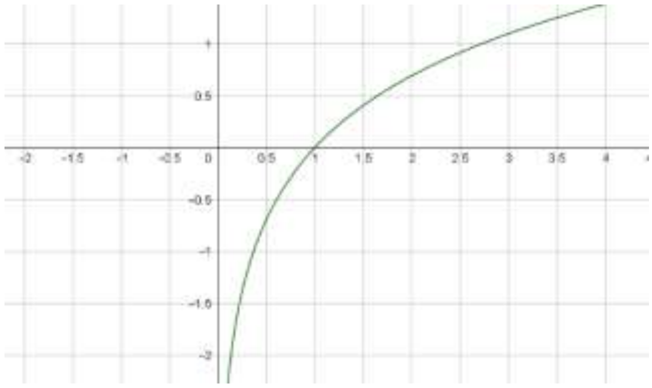
```
def brouncker(n):
    S = 0
    for k in range(n+1):
        S =
    return
```

[Retour présentation algorithmes](#)

L'algorithme de Briggs

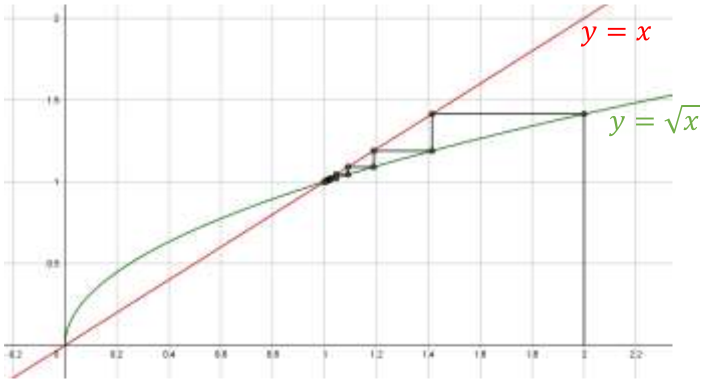
Henry Briggs est un mathématicien anglais né en 1556 et mort en 1630. En 1624, 10 ans après la « découverte » des logarithmes par John Neper, il publie son ouvrage *Arithmetica logarithmica* dans lequel on trouve des tables très précises du logarithme décimal.

1. Déterminer une équation de la tangente à la courbe de la fonction ln au point d'abscisse 1.
2. Tracer cette tangente sur le graphique ci-dessous :



Ainsi pour u « proche » de 1, on a $\ln(u) \approx u - 1$. On pourra faire percevoir aux élèves le lien avec $\lim\limits_{u \to 1} \frac{\ln(u)}{u-1} = 1$.

3. On considère la suite définie par $u_0 = 2$ et pour tout entier n, $u_{n+1} = \sqrt{u_n}$.
   a) Compléter la représentation ci-dessous pour faire apparaître u_0, u_1 et u_2 sur l'axe des abscisses.



   b) Conjecturer le comportement de la suite (u_n) (variation et convergence éventuelle).

Dans la suite de l'exercice, on admet que la suite (u_n) est décroissante et converge vers 1 (ces affirmations pourraient être démontrées).

4. Montrer par récurrence que pour tout entier naturel n, $2^n \ln(u_n) = \ln(2)$. On admet donc que, pour n « suffisamment » grand, ln$(2) \approx 2^n(u_n - 1)$.
5. Compléter les fonctions suivantes écrites en Python. La première correspond à l'algorithme de Briggs qui renvoie une valeur approchée de $\ln(2)$. La deuxième renvoie le nombre d'étapes nécessaires afin d'obtenir une valeur approchée à 10^{-p} de $\ln(2)$ (qui se note log(2) en Python).

```
def briggs(p):
    n = 0
    u=2
    while u-1>10**(-p):
        ………………..
        ………………..
    return ………….
```

```
def etape_briggs(p):
    n =
    u =
    while abs(……….-log(2)) > …………….  :
    u =
    n =
    return n
```

Retour présentation algorithmes