**Fonctions élémentaires autour de la dérivation**

***But de l’activité :*** *Ecrire des fonctions Python permettant le calcul de taux de variation, de nombres dérivés, du coefficient directeur et de l’ordonnée à l’origine d’une tangente à une courbe.*

On considère la fonction *f* définie sur par .

1. Ecrire une fonction Python **f** qui :  
   - reçoit en argument une valeur

- renvoie son image par la fonction .

1. Ecrire une fonction Python **coeff\_dir** qui :  
   - reçoit en arguments les coordonnées de deux points et (avec )

- renvoie le coefficient directeur de la droite .

1. A l’aide de la fonction précédente, écrire une fonction Python **taux\_variation** qui :  
   - reçoit en arguments une fonction et deux valeurs et   
   - renvoie le taux de variation de la fonction entre et .
2. A l’aide de cette fonction, calculer le taux de variation de entre 3 et 3,000001.

Conjecturer la valeur du nombre dérivé , puis effectuer un calcul pour vérifier.

1. L’import « from scipy import misc » permet d’utiliser la fonction **misc.derivative** qui :  
   - reçoit en arguments une fonction et une valeur

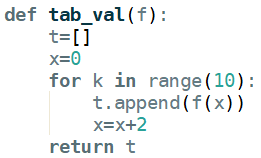
- renvoie le nombre dérivé de en .

Tester cette fonction pour calculer .

1. Ecrire une fonction Python **coeff\_tang** qui :  
   - reçoit en arguments une fonction et une valeur   
   - renvoie le coefficient directeur et l’ordonnée à l’origine de la tangente à en .

Tester cette fonction pour déterminer l’équation de la tangente à la courbe de en 2.

1. La fonction **tab\_val** ci-contre permet d’obtenir une liste de valeurs de la fonction  :



**a)** Quelle est la valeur initiale de cette liste ? le pas ? le nombre de valeurs obtenues ?  
**b)** Adapter cette fonction pour qu’elle reçoive en argument la valeur initiale , le pas et le nombre de valeurs .

1. Ecrire une fonction Python cdir\_secantes qui :

- reçoit en arguments une fonction , une valeur , un pas et un entier .

- renvoie la liste des coefficients directeurs des sécantes à la courbe de à partir de avec un pas en abscisse .

**Méthode de Newton**

***Prérequis :*** *Fonctions Pythons réalisées dans l’activité « Fonctions élémentaires autour de la dérivation »****But de l’activité :*** *Approcher la solution d’une équation à l’aide de la méthode de Newton.*

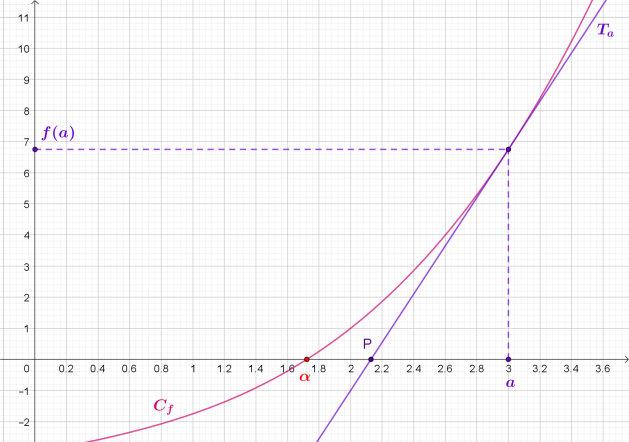
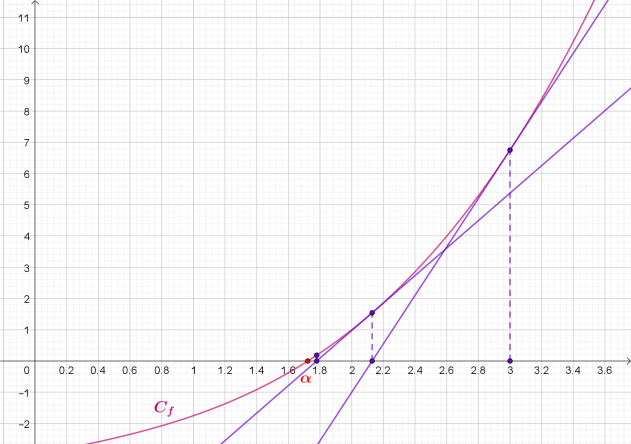
On considère la fonction *f* définie sur par .

1. Démontrer que est croissante sur .  
   *On admettra pour la suite que l’équation a une unique solution sur , notée .*
2. Justifier que pour toute abscisse , la tangente à la courbe de en coupe l’axe des abscisses en un point .

Déterminer l’expression de l’abscisse de en fonction de , et .

Ecrire une fonction Python **etap\_Newton** qui :  
- reçoit en argument une fonction et une valeur

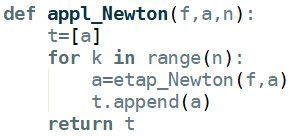
- renvoie l’abscisse du point correspondant

**Figure pour la question 2 Figure pour la question 3**

1. *A partir d’un point de l’axe des abscisses, on peut donc construire une suite de points.  
   On admettra ici que la suite des abscisses de ces points a pour limite .*

**a)** La fonction Python **appl\_Newton** donnée ci-contre :  
 - reçoit en arguments une fonction , une valeur et un entier *n*



**-** renvoie une liste de valeurs.   
 Expliquer ce que représentent les termes de la liste renvoyée.

**b)** Coder cette fonction et tester pour la fonction de l’énoncé avec et .

1. **a)** Proposer et coder en Python des fonctions et s’annulant respectivement en et .

**b)** A l’aide des fonctions Python précédentes, proposer des valeurs approchées de ces deux nombres.

**Algorithme de dichotomie**

***Prérequis :*** *Aucun, mais les question 1)a)b) peuvent être supprimées si l’activité « Méthode de Newton » a été traitée.*  
***But de l’activité :*** *Approcher la solution d’une équation à l’aide d’un algorithme de dichotomie (méthode plus lente que la méthode de Newton, mais pour laquelle la précision du résultat est connue).*   
On considère la fonction *f* définie sur par .

1. **a)** Démontrer que est croissante sur

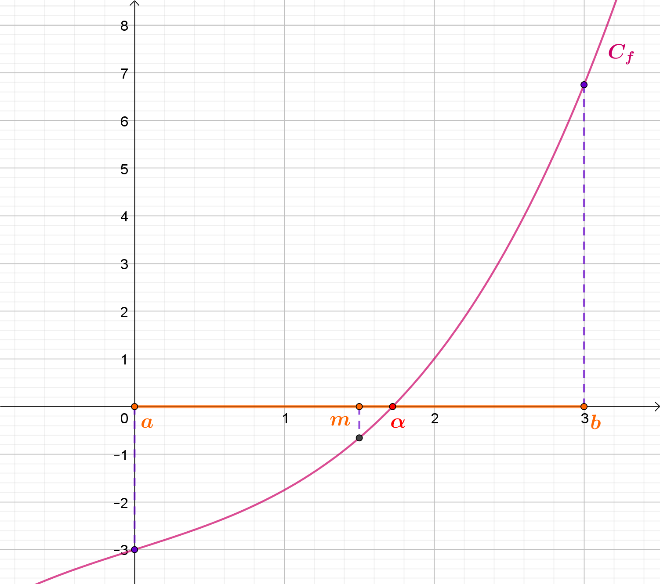
*On admettra pour la suite que l’équation a une unique solution sur , notée*

**b)** Ecrire une fonction Python **f** qui :  
 - reçoit en argument une valeur   
 - renvoie son image par la fonction .

**c)** Déterminer les images de 0 et 3 par , et en déduire que *.*

1. **a)** On considère un intervallecontenantet on pose.

Justifier que : **(\*) sialors *,* et sinon**

****b)** En utilisant **(\*)**, écrire une fonction Python **etap\_dichoto** qui :  
 - reçoit en arguments une fonction *f* et les   
 bornes et d’un intervalle contenant   
 - renvoie les bornes et d’un nouvel   
 intervalle contenant .

**c)** A partir de l’intervalle , obtenir successivement 3 nouveaux intervalles contenant .

**d)** Que peut-on dire de la longueur de chaque intervalle obtenu par rapport à la précédente ?

1. **a)** Ecrire une fonction Python **dichoto\_iter** qui :  
    - reçoit en arguments une fonction *f* , les bornes et d’un intervalle contenant et un   
    entier   
    - renvoie les bornes d’un nouvel intervalle contenant obtenu en répétant fois la   
    fonction précédente.

**b)** Tester avec la fonction de l’énoncé en partant de l’intervalle et en répétant 10 fois la méthode.

1. **a)** Ecrire une fonction Python **dichoto\_test** qui :  
    - reçoit en argumentsla fonction *f*, les bornes et d’un intervalle contenant et une

valeur

**-** renvoie les bornes du premier intervalle de longueur inférieure à obtenu avec la   
 méthode décrite précédemment.

**b)** Tester avec la fonction de l’énoncé pour obtenir un encadrement de à près.

1. **a)** Proposer et coder en Python des fonctions et s’annulant respectivement en et .

**b)** A l’aide des fonctions Python précédentes, proposer des encadrements de ces deux nombres à près.