Situation 1

Compléter le script proposé ci-contre afin qu'il affiche le nombre dérivé de la fonction carrée au point d'abscisse 3. Modifier ce script afin qu'il affiche le nombre dérivé de la fonction inverse au point d'abscisse 3

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

def nbederive(f,a):
  h=1e-10
  v=(...-...)/h
  return(v)

print(nbederive(f,3))
```

Situation 2

On considère la fonction f définie par $f(x)=x^2-2$ qui s'annule pour $x=\sqrt{2}$. On souhaite déterminer une valeur approchée de la racine de cette fonction à l'aide de la méthode de Newton proposée en page 10 des activités.

Compléter le script proposé ci-contre afin d'être en mesure de comparer une valeur approchée de $\sqrt{2}$ après 10 itérations de la méthode de Newton et la valeur affichée par la commande « sqrt(2) ». Commenter…

```
from math import *

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

def nbederive(f,a):
  h=1e-10
  v=(...-...)/h
  return(v)

def newton(n):
  x=2
  for i in range(...):
    x=...
  return(x)

print(newton(10))
print(sqrt(2))
```

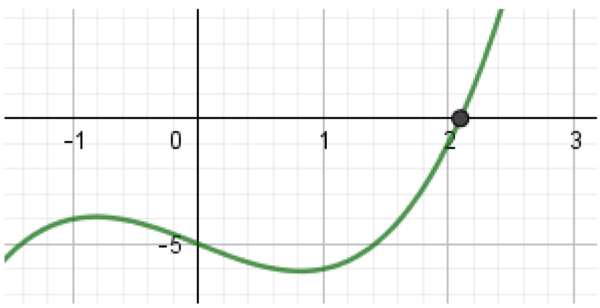
Situation 3

On reprend les données de la situation 2. Compléter les cellules d'un tableur afin de déterminer une valeur approchée de $\sqrt{2}$ à l'aide de la méthode de Newton. Commenter les résultats obtenus en termes de vitesse de convergence.

	A	B	C	D
1	INDICE	ABSCISSE	IMAGE	NBE DERIVE
2	0	2	2	4
3	1	1,5	0,25	3
4	2	1,416666667	0,006944444	2,833333333
5	3	1,414215686	6,0073E-06	2,828431373
6	4	1,414213562	4,51061E-12	2,828427125

Situation 4

On considère la fonction définie par $f(x)=x^3-2x-5$. On s'intéresse à la racine de cette fonction, c'est-à-dire au nombre α tel que $f(\alpha)=0$. Compléter le script pour obtenir une valeur approchée de α par la méthode de Newton.




```
def f(x):
  return(...)

def fprime(x):
  h=...
  v=(...-...)/h
  return(...)

def newton(x,n):
  l=[]
  l.append(x)
  for i in range(1,n+1):
    x=l[i-1]
    l.append(...-.../...)
  return(...)

def affichage(l):
  n=len(l)
  for k in range(n):
    print("x{}={}".format(k,l[k]))

affichage(newton(1.5,5))
```

<u>Situation 5</u>

On reprend les données de la situation 4. Compléter les cellules du tableur afin de déterminer une valeur approchée de α à l'aide de la méthode de Newton. Cette même équation a été étudiée par Isaac Newton dans son livre « La méthode des fluxions et les suites infinies » (1736)…

	A	B	C	D
1	INDICE	ABSCISSE	IMAGE	NBE DERIVE
2	0	1,5	-4,625	4,75
3	1	2,473684211	5,189386208	16,35734072
4	2	2,156432996	0,714985771	11,9506098
5	3	2,096604604	0,022942293	11,18725259
6	4	2,094553851	2,64437E-05	11,1614675
7	5	2,094551482	3,52705E-11	11,16143773

<u>Situation 6</u>

Le mathématicien Euler affirmait : si h est **« petit »** alors : $\boxed{f(a+h) \approx f(a)+hf'(a)}$ $\langle Euler \rangle$. Nous appellerons « relation d'Euler » cette approximation. Comment expliquez-vous sa validité ?

On s'intéresse ici à la représentation graphique sur l'intervalle $[-2;2]$ d'une fonction f prenant la valeur 1 en 0 et dont le nombre dérivé en chaque point est égal à son image. C'est-à-dire une fonction qui vérifie les deux conditions suivantes : $f(0)=1$ et $f'(x)=f(x)$ pour tout x réel.

La relation d'Euler pour un h petit et positif

1. Ecrire la relation $\langle Euler \rangle$ pour $a=0$ et $h=0{,}1$. Déterminer $f(0{,}1)$.
2. Ecrire la relation $\langle Euler \rangle$ pour $a=0{,}1$ et $h=0{,}1$. Déterminer $f(0{,}2)$.
3. Ecrire la relation $\langle Euler \rangle$ pour $a=0{,}2$ et $h=0{,}1$. Déterminer $f(0{,}3)$ à 10^{-2} près.
4. Continuer le travail… Compléter le script afin de tracer la partie positive de la courbe.

La relation d'Euler pour un h petit et négatif

5. Ecrire la relation $\langle Euler \rangle$ pour $a=0$ et $h=-0{,}1$. Déterminer $f(-0{,}1)$.
6. Ecrire la relation $\langle Euler \rangle$ pour $a=-0{,}1$ et $h=-0{,}1$. Déterminer $f(-0{,}2)$.
7. Ecrire la relation $\langle Euler \rangle$ pour $a=-0{,}2$ et $h=-0{,}1$. Déterminer $f(-0{,}3)$ à 10^{-2} près.
8. Continuer le travail… Compléter le script afin de tracer la partie négative de la courbe.

```python
from math import *
import matplotlib.pyplot as plt

h=...
abs=[]
ord=[]
abs.append(...)
ord.append(...)

for i in range(...):
  abs.append(...)
  ord.append(...)

plt.plot(abs,ord)
plt.show()
```