exercice 1

1)

Entrée [34]:

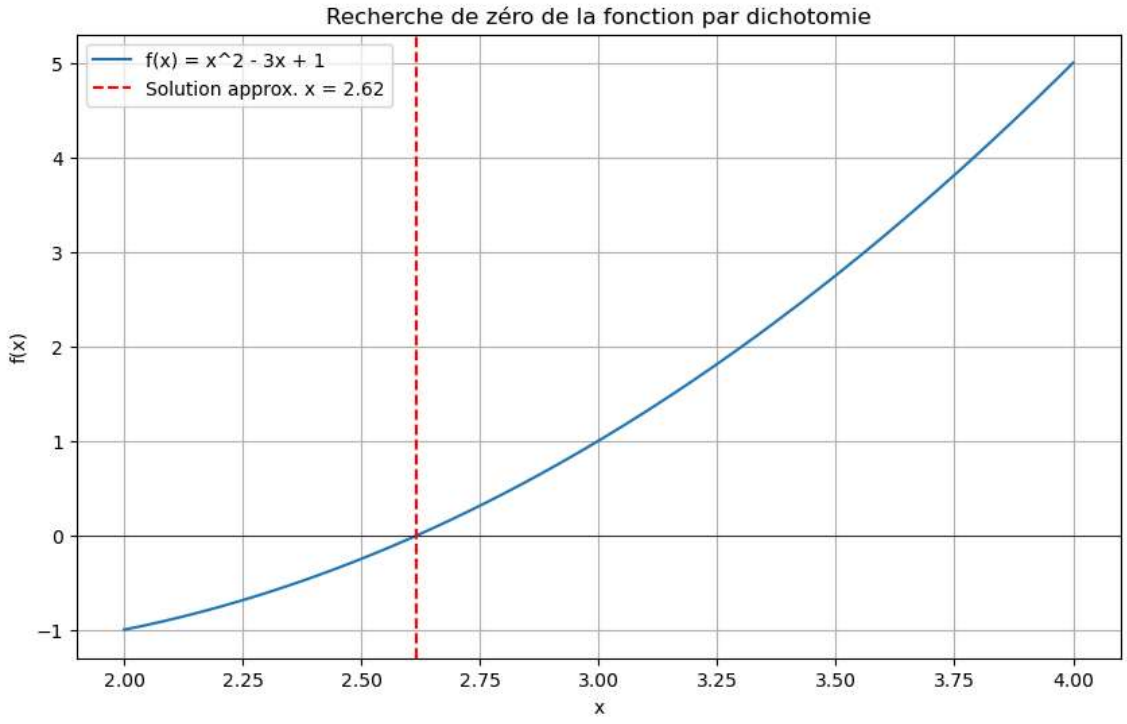
```
def f(x):
    return x**2 - 3*x + 1

def dichotomie(a, b, precision):
    trace = []
    while (b - a) / 2 > precision:
        c = (a + b) / 2
        trace.append((a, b, b - a, c, f(c)))
        if f(c) == 0:
            return c, trace
        elif f(a) * f(c) < 0:
            b = c
        else:
            a = c
    trace.append((a, b, b - a, (a + b) / 2, f((a + b) / 2)))
    return (a + b) / 2, trace

# Intervalle initial
a = 2
b = 4
precision = 0.01



# Créer un tableau des valeurs intermédiaires
columns = ['Borne a', 'Borne b', 'Amplitude (b-a)', 'Centre', 'Image du cer
df = pd.DataFrame(trace, columns=columns)
df
```

Out[34]:

	Borne a	Borne b	Amplitude (b-a)	Centre	Image du centre
0	2.000000	4.000	2.000000	3.000000	1.000000
1	2.000000	3.000	1.000000	2.500000	-0.250000
2	2.500000	3.000	0.500000	2.750000	0.312500
3	2.500000	2.750	0.250000	2.625000	0.015625
4	2.500000	2.625	0.125000	2.562500	-0.121094
5	2.562500	2.625	0.062500	2.593750	-0.053711
6	2.593750	2.625	0.031250	2.609375	-0.019287
7	2.609375	2.625	0.015625	2.617188	-0.001892

2)

Entrée [35]:
```
import matplotlib.pyplot as plt
import numpy as np

# Définir la fonction
def f(x):
    return x**2 - 3*x + 1

# Définir l'intervalle et les points
x_values = np.linspace(2, 4, 400)
y_values = f(x_values)

# Tracer la fonction
plt.figure(figsize=(10, 6))
plt.plot(x_values, y_values, label='f(x) = x^2 - 3x + 1')
plt.axhline(0, color='black', linewidth=0.5)
plt.axvline(solution, color='red', linestyle='--', label=f'Solution approx.
plt.xlabel('x')
plt.ylabel('f(x)')
plt.title('Recherche de zéro de la fonction par dichotomie')
plt.legend()
plt.grid(True)
plt.show()
```



Entrée [33]:
```
# Trouver la solution et les valeurs intermédiaires
solution, trace = dichotomie(a, b, precision)

import pandas as pd
```

Out[33]: 2.6171875

3)

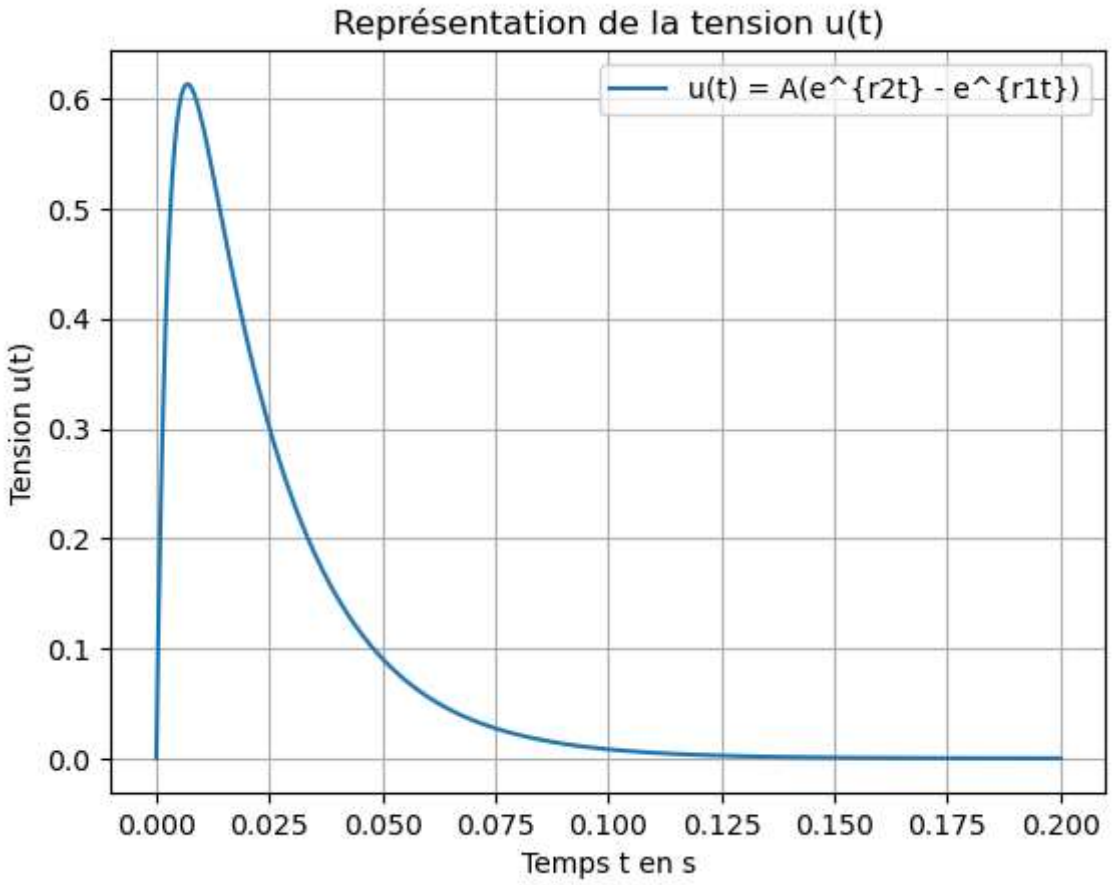
Entrée [23]:
```
# Créer un tableau des valeurs intermédiaires
columns = ['Borne a', 'Borne b', 'Amplitude (b-a)', 'Centre', 'Image du cer
df = pd.DataFrame(trace, columns=columns)
df
```

Out[23]:

	Borne a	Borne b	Amplitude (b-a)	Centre	Image du centre
0	2.000000	4.000	2.000000	3.000000	1.000000
1	2.000000	3.000	1.000000	2.500000	-0.250000
2	2.500000	3.000	0.500000	2.750000	0.312500
3	2.500000	2.750	0.250000	2.625000	0.015625
4	2.500000	2.625	0.125000	2.562500	-0.121094
5	2.562500	2.625	0.062500	2.593750	-0.053711
6	2.593750	2.625	0.031250	2.609375	-0.019287
7	2.609375	2.625	0.015625	2.617188	-0.001892

Exercice 2

Entrée [36]:
```
import numpy as np
import matplotlib.pyplot as plt

# Définir les constantes
A = 1  # Vous pouvez ajuster cette valeur selon les données du TD
r1 = -327  # en s^-1
r2 = -48   # en s^-1

# Définir la fonction u(t)
def u(t):
    return A * (np.exp(r2 * t) - np.exp(r1 * t))

# Générer des valeurs de t
t = np.linspace(0, 0.2, 400)

# Calculer u(t) pour chaque t
u_t = u(t)

# Tracer la courbe
plt.plot(t, u_t, label='u(t) = A(e^{r2t} - e^{r1t})')
plt.xlabel('Temps t en s')
plt.ylabel('Tension u(t)')
plt.title('Représentation de la tension u(t)')
plt.legend()
plt.grid(True)
plt.show()
```



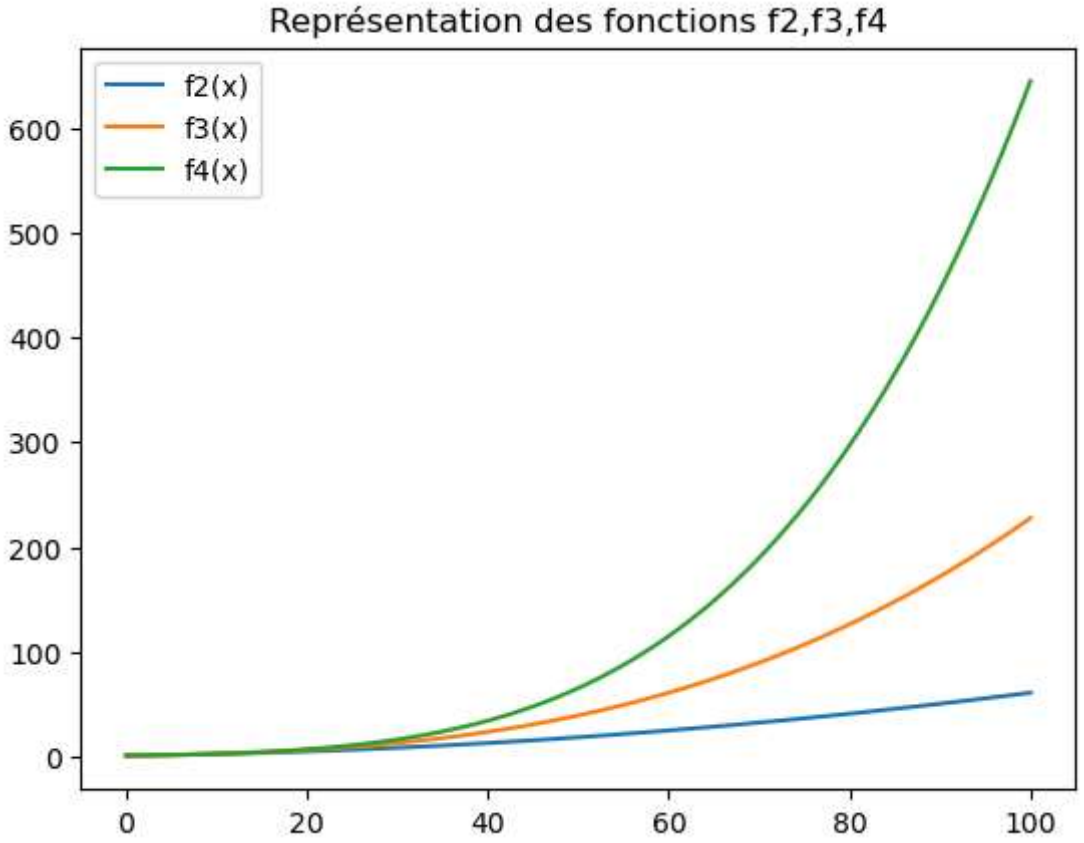
Exercice 3

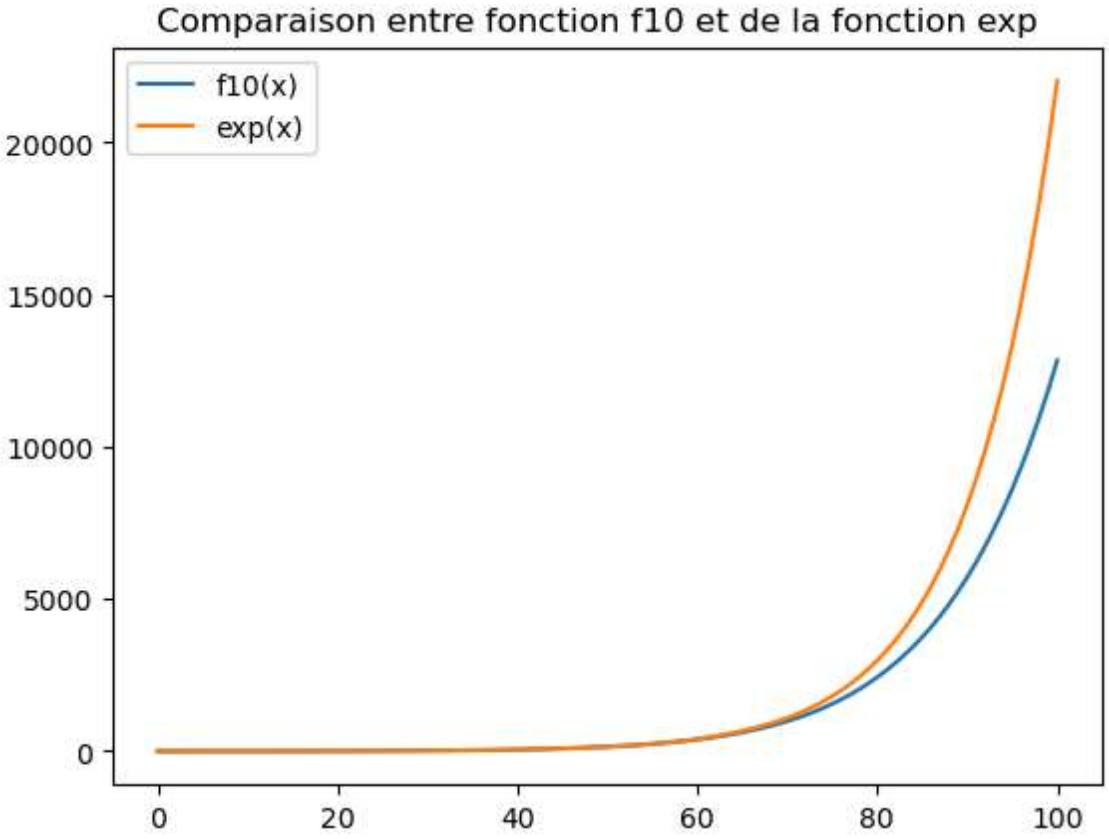
Entrée [41]:

```
# Définition de la fonction factoriel
def factoriel(k):
    n = k
    if n >= 0:
        if n == 0 or n == 1:
            return 1
        else:
            return n * factoriel(n-1)
    else:
        return float('NaN')


# Définition de la fonction S

def S(x, n):
    somme = 0
    for k in range(n+1):
        terme = x**k / factoriel(k)
        somme += terme
    return somme
```

Entrée [43]:

```
import matplotlib.pyplot as plt
# Représentation des fonctions f2,f3,f4,
x = list(range(101))
y2 = [S(i/10, 2) for i in x]
y3 = [S(i/10, 3) for i in x]
y4 = [S(i/10, 4) for i in x]

plt.plot(x, y2, label='f2(x)')
plt.plot(x, y3, label='f3(x)')
plt.plot(x, y4, label='f4(x)')
plt.legend()
plt.title('Représentation des fonctions f2,f3,f4')
plt.show()
import math
# Comparaison de la fonction f10 et exp
x = list(range(101))
y10 = [S(i/10, 10) for i in x]
yexp = [math.exp(i/10) for i in x]

plt.plot(x, y10, label='f10(x)')
plt.plot(x, yexp, label='exp(x)')
plt.legend()
plt.title('Comparaison entre fonction f10 et de la fonction exp  ')
plt.show()
```




Comparaison entre fonction f10 et de la fonction exp
f10(x)
exp(x)
20000
15000
10000
5000
0
0
20
40
60
80
100

Entrée [ ]: