```
# E:\TP info 24-25\TP2 Exercices 1 à 6.py
# Exercice 1
import matplotlib.pyplot as plt
import numpy as np
## 1
def f1(x):
    return x**2-x+1
Lx = np.linspace(0,1)
Ly = [f1(x) for x in Lx]
# Ly = f1(Lx)
plt.plot(Lx,Ly)
plt.show()

## 2
def f2(x):
    return x*np.cos(2*np.pi*x)
Lx = np.linspace(-1,1)
Ly = [f2(x) for x in Lx]
plt.plot(Lx,Ly)
plt.show()

## 3
def f3(x):
    return x**2-2
Lx = np.linspace(1,2)
Ly = [f3(x) for x in Lx]
plt.plot(Lx,Ly)
plt.show()

## 4
def f4(x):
    return 1/(x**2 + 1)
Lx = np.linspace(0,1)
Ly = [f4(x) for x in Lx]
plt.plot(Lx,Ly)
plt.show()


## 5
def f5(x):
    return x*np.exp(-x)
Lx = np.linspace(-1,1,200)
Ly = [f5(x) for x in Lx]
plt.plot(Lx,Ly)
plt.show()


## 6
def f6(x):
    return np.log(1+x)
```

```python
Lx = np.linspace(1,2)
Ly = [f6(x) for x in Lx]
plt.plot(Lx,Ly)
plt.show()

## Exercice 2
def Riemann(f,a,b,n):
    h = (b-a)/n
    S = 0
    for k in range(n):
        S += f(a+k*h)
    return h*S

def f(t):
    return 4/(1+t**2)

a, b = 0, 1
n = 1000
print(Riemann(f,a,b,n))

def best_n(p):
    a, b = 0, 1
    n = 1
    while abs(Riemann(f,a,b,n)-np.pi) > 10**(-p):
        n += 1
    return n

print(best_n(3))

for p in range(1,5):
    print(best_n(p))
##
Lp = range(1,10)
Lbest = [best_n(p) for p in Lp]
plt.plot(Lp,Lbest)
plt.show()

# Exercice 3
## 1
def f1(x):
    return 2-x**2

a, b = 0, 2
n = 1000
print(Riemann(f1,a,b,n))

## 2
def f2(x):
    return np.log(x) - 1

```

```
a, b = 1, 2
n = 1000
print(Riemann(f2,a,b,n))

## 3
def f3(x):
    return np.sqrt(x) - 2

a, b = 1, 6
n = 1000
print(Riemann(f3,a,b,n))

## 4
def f4(x):
    return x - 2*np.sin(2*x)

a, b = 0, 3
n = 100
print(Riemann(f4,a,b,n))

## Exercice 4
def u(n):
    def f(x):
        return x**2 * np.sin(n*x)
    return Riemann(f,0,1,100)

def liste(n):
    return [u(k) for k in range(n+1)]

n = 100
Ln = range(n+1)
Lu = liste(n)
plt.plot(Ln,Lu)
plt.show()

# on conjecture que les deux suites extraites sont adjacentes
# donc, que la suite est convergente

# Exercice 5
## 1
def F1(x):
    def f(t):
        return np.exp(-t*x)/(1+t)
    return Riemann(f,0,1,100)

Lx = np.linspace(1,5,500)
Ly = [F1(x) for x in Lx]
plt.plot(Lx,Ly)
plt.show()
```

```python

## 2
def F2(x):
    def f(t):
        return np.sin(x+t)/(x+t)
    return Riemann(f,0,1,100)

Lx = np.linspace(2,4,200)
Ly = [F2(x) for x in Lx]
plt.plot(Lx,Ly)
plt.show()

## 3
def F3(x):
    def f(t):
        return np.exp(-x*(1+t**2))/(1+t**2)
    return Riemann(f,0,1,100)

Lx = np.linspace(-1,1,200)
Ly = [F3(x) for x in Lx]
plt.plot(Lx,Ly)
plt.show()

## Exercice 6
def Rectangles(a,b,h,f):
    n = int(np.floor((b-a)/h))
    return Riemann(f,a,b,n)

def f(t):
    return 4/(1+t**2)

print(Rectangles(0,1,0.001,f))

def rectangles(a,b,h,f):
    L = np.arange(a,b,h)
    S = 0
    for el in L:
        S += f(el)
    return h*S

print(rectangles(0,1,0.001,f))

## Exercice 7
# y(x) = exp(-x)
import numpy as np
import matplotlib.pyplot as plt
a, b = 0, 1
h = 0.2
Lx = np.arange(a,b,h)
n = int(np.floor((b-a)/h))
```

```
Ly = [1]
for k in range(n-1):
    Ly.append((1-h)*Ly[-1])
print(Ly)

for p in [1,2]:
    h = 10**(-p)
    Lx = np.arange(a,b,h)
    n = int(np.floor((b-a)/h))
    Ly = [1]
    for k in range(n-1):
        Ly.append((1-h)*Ly[-1])
    plt.plot(Lx,Ly)
Lx = np.linspace(0,1)
Ly = [np.exp(-x) for x in Lx]
plt.plot(Lx,Ly)
plt.show()

## Exercice 8
import numpy as np
import matplotlib.pyplot as plt
def F(x,y):
    return -2*x*y+x*np.exp(-x**2)*y**3

a, b = 0, 2
h = 0.02
Lx = np.arange(a,b,h)
n = int(np.floor((b-a)/h))
Ly = [1]
for k in range(n-1):
    Ly.append((Ly[k]+h*F(Lx[k],Ly[k])))
plt.plot(Lx,Ly)

def f(x):
    return np.sqrt(3/(np.exp(-x**2)+2*np.exp(2*x**2)))
Lexact = f(Lx)
plt.plot(Lx,Lexact)
plt.show()

## Exercice 9
import numpy as np
import matplotlib.pyplot as plt
def F(x,y):
    return -(2/x)*y+x**2

a, b = 0.05, 2
h = 0.025
Lx = np.arange(a,b,h)
n = int(np.floor((b-a)/h))
Ly = [2/5]
```

```
for k in range(n-1):
    Ly.append((Ly[k]+h*F(Lx[k],Ly[k])))
plt.plot(Lx,Ly,'black')

def f(x):
    return 0.001/x**2+(x**3)/5
Lexact = f(Lx)
plt.plot(Lx,Lexact,'red')
plt.show()

##
a, b = 1, 200
h = 5
Lx = np.arange(a,b,h)
n = int(np.floor((b-a)/h))
Ly = [2/5]
for k in range(n-1):
    Ly.append((Ly[k]+h*F(Lx[k],Ly[k])))
plt.plot(Lx,Ly,'black')

def f(x):
    return 0.001/x**2+(x**3)/5
Lexact = f(Lx)
plt.plot(Lx,Lexact,'red')
plt.show()
```