1. MÉTHODE DES RECTANGLES À GAUCHE

Corrigé I1

Calcul de l'intégrale :

$$\int_0^{3\pi/2}\cos(x)dx=\left[\sin(x)\right]_0^{3\pi/2}=\sin\left(\frac{3\pi}{2}\right)-son(0)=-1-0=-1$$

Les résultats précédents sont donc approchés, ce qui était prévu.

Fonction associée à la méthode : ***version améliorée + test validité***

```
"""
intégration numérique par la méthode des rectangles à gauche : fonction améliorée + test
"""
import numpy as np

# intégration numérique
def rectanglesG(a,b,n,f):
    S=0
    for i in range(n-1):
        x1 = a+i*(b-a)/n
        x2 = a+(i+1)*(b-a)/n
        S = S + f(x1)*(x2-x1)
    return S

# application
def f(x) :
    return np.cos(x)
a = 0
b = 3*np.pi/2
n = 2000          # on augmente le nombre d'intervalles pour affiner le calcul
I = rectanglesG(a,b,n,f)
print('valeur de l''intégrale =', I)

# validité
ecart = -(I+1)              # écart relatif = (résultat trouvé - val théo)/val théo
print('écart relatif (avec',f'n={n}) =', ecart*100, '%')
            # on peut indiquer la valeur de n et multiplier par 100 pour avoir des %
```

2. VARIANTES : MÉTHODE DES RECTANGLES À DROITE, MÉTHODE DES RECTANGLES MOYENS

Corrigé I4

```
"""
intégration numérique par la méthode des rectangles moyens : visualisation
"""
import numpy as np
import matplotlib.pyplot as plt

#paramètres de l'intégration
a = 0
b = 3*np.pi/2
n = 20

# définition et tracé de la fonction
def f(x):
    return np.cos(x)
x = np.linspace(a, b, n)
plt.plot(x,f(x),'r')

# intégration numérique
S = 0
for i in range(n-1):
    S = S + (f(x[i]+f(x[i+1])/2*(x[i+1]-x[i])
    # dessin du rectangle
    x_rect = [x[i], x[i]           , x[i+1]            , x[i+1], x[i]] # abscisses des sommets
    y_rect = [0   , f(x[i]+x[i+1])/2, f(x[i]+x[i+1])/2, 0     , 0   ] # ordonnées des sommets
    plt.plot(x_rect, y_rect,'orange')
print('valeur de l''intégrale numérique =', S)
plt.show()
```

Corrigé I5

```
"""
intégration numérique par la méthode des rectangles moyens : fonction
"""
import numpy as np

# intégration numérique
def rectanglesM(a,b,n,f):
    S=0
    for i in range(n-1):
        x1 = a+i*(b-a)/n
        x2 = a+(i+1)*(b-a)/n
        S = S + f((x1+x2)/2)*(x2-x1)
    return S

# application
def f(x) :
    return np.cos(x)
a = 0
b = 3*np.pi/2
n = 2000
I = rectanglesM(a,b,n,f)
print('valeur de l''intégrale =', I)
```

3. MÉTHODE DES TRAPÈZES

Corrigé I6

```
"""
intégration numérique par la méthode des trapèzes : visualisation
"""
import numpy as np
import matplotlib.pyplot as plt

#paramètres de l'intégration
a = 0
b = 3*np.pi/2
n = 20

# définition et tracé de la fonction
def f(x):
    return np.cos(x)
x = np.linspace(a, b, n)
plt.plot(x,f(x),'r')

# intégration numérique
S = 0
for i in range(n-1):
    S = S + (x[i+1]-x[i])/2*(f(x[i]+f(x[i+1])
    # dessin du trapèze
    x_trap = [x[i], x[i]    , x[i+1]      , x[i+1], x[i]] # abscisses des sommets
    y_trap = [0    , f(x[i]), f(x[i+1])   , 0      , 0    ] # ordonnées des sommets
    plt.plot(x_trap, y_trap,'y')
print('valeur de l''intégrale numérique =', S)

plt.show()
```

Corrigé I7

```
"""
intégration numérique par la méthode des trapèzes : fonction
"""
import numpy as np

# intégration numérique
def trapezes(a,b,n,f):
    S=0
    for i in range(n-1):
        x1 = a+i*(b-a)/n
        x2 = a+(i+1)*(b-a)/n
        S = S + (x2-x1)/2 * (f(x1)+f(x2))
    return S

# application
def f(x) :
    return np.cos(x)
a = 0
b = 3*np.pi/2
n = 2000
I = trapeze(a,b,n,f)
print('valeur de l''intégrale =', I)
```

4. APPLICATION

Corrigé I8

```
"""
valeur moyenne de cos² par la méthode des rectangles moyens
"""
import numpy as np

# intégration numérique
def rectanglesM(a,b,n,f):
    S=0
    for i in range(n-1):
        x1 = a+i*(b-a)/n
        x2 = a+(i+1)*(b-a)/n
        S = S + f((x1+x2)/2)*(x2-x1)
    return S

# application
def f(x) :
    return np.cos(x)
T = 3*np.pi/2
n = 1000000
I = rectanglesM(0,T,n,f)
M = I/T
print('valeur moyenne de cos² =', M)

# validité
ecart = 2*(M-0.5)
print('écart relatif (avec',f'n={n}) =', ecart*100, '%')
```

On constate que les méthodes des rectangles moyens et des trapèzes ont sensiblement la même précision.