

```
# -*- coding: utf-8 -*-  
"""
```

```
Created on Thu Feb 13 01:38:28 2014
```

```
@author: dconduche  
"""
```

```
"""
```

```
Les remarques sont identiques à celles pour les rectangles :  
seule la formule change, et encore, à la marge seulement.  
"""
```

```
import numpy as np
```

```
def trapeze(f, a, b, n):  
    """Calcule l'intégrale de f sur [a,b] par la méthode des trapèzes.
```

```
    Keyword arguments:
```

```
    f -- fonction continue par morceaux définie sur [a,b]
```

```
    a -- float représentant le début de l'intervalle
```

```
    b -- float représentant la fin de l'intervalle (a<b et f(a)f(b)<0)
```

```
    n -- int représentant le nombre de rectangles utilisés
```

```
    retourne un float, l'intégrale de f sur [a,b] avec n pas.
```

```
    """
```

```
    I = (f(a) + f(b)) / 2
```

```
    pas = (b-a) / n
```

```
    for k in range(1, n):
```

```
        I = I + f(a + k*pas)
```

```
    return pas * I
```

```
def trapezeBis(f, a, b, n):
```

```
    pas = (b-a) / n
```

```
    res = (f(a) + f(b))/2 + sum(f(a + k*pas) for k in range(1, n))
```

```
    return pas * res
```

```
def trapezeNP(f, a, b, n):
```

```
    x = np.linspace(a, b, n+1)[1:-1]
```

```
    pas = (b-a) / n
```

```
    res = (f(a) + f(b))/2 + sum(f(x))
```

```
    return pas * res
```

```
f = lambda x: x**2
```

```
I = trapeze(f, 0, 1, 100000)
```

```
Inp = trapezeNP(f, 0, 1, 100000)
```

```
print(I, Inp) # Calcul exact : 1/3
```