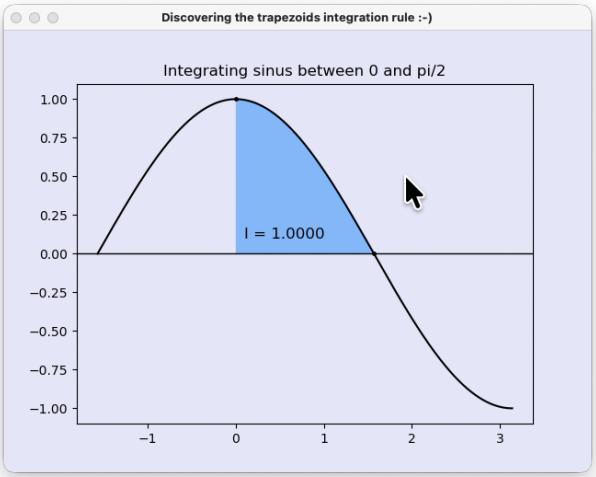
Python for dummies : problème 5
Des trapèzes, des trapèzes...
plein de trapèzes !



Le devoir de la semaine passée était très très compliqué, l'enseignant doit donc se faire pardonner et vous a concocté un problème très très simple :-)
Il vous demande seulement d'implémenter la règle composite d'intégration des trapèzes.

On partage l'intervalle d'intégration $[a, b] = [X_0, X_n]$ en n sous-intervalles égaux. La longueur de chaque sous-intervalle vaudra h qui est donc l'écart entre deux abscisses voisines d'intégration X_i et X_{i+1}.

La règle composite des trapèzes s'écrit donc :

$$\int_a^b u(x)\,dx \quad = \quad I \quad \approx \quad I^n \quad = \quad \frac{h}{2}\left(U_0 + 2U_1 + 2U_2 + \ldots + 2U_{n-1} + U_n\right)$$

Plus précisément, on vous demande de :

1. Ecrire une fonction `I = trapezeEasy(f,a,b,n)` qui calcule l'intégrale de la fonction `f` entre les abscisses `a` et `b` en utilisant la règle composite des trapèzes avec `n` intervalles. Il y a donc bien $n+1$ points où on évalue la fonction. Comme le nom de la fonction l'indique, cette partie du devoir est vraiment très simple : tous les étudiants sont donc vraiment invités à réaliser au minimum cette toute petite contribution. Oui, oui : tous les étudiants même ceux qui estiment que toucher le clavier d'un ordinateur est indigne de leur futur statut de cadre supérieur ou de dirigeant d'entreprise ! On peut supposer que la fonction `f` peut prendre un tableau comme argument et restituer un tableau de même taille comme output.

2. Ecrire une fonction `I,n,errorEst = trapezeFun(f,a,b,n,nmax,tol)` qui calcule à nouveau l'intégrale de la fonction `f` entre les abscisses `a` et `b` en utilisant la règle composite des trapèzes avec `n` intervalles[1] Ensuite, cette fonction répétera le calcul en multipliant successivement le nombre d'intervalle par deux jusqu'à obtenir une précision inférieure à une tolérance ϵ qui sera toujours un nombre réel strictement positif défini par l'argument `tol`. Pour estimer l'erreur d'intégration, on comparera la valeur obtenue I_{2n} avec celle de I_n et on arrêtera le calcul lorsque

   $$|I_{2n} - I_n| < \epsilon$$

   .

   Afin d'éviter de se perdre dans les affres d'un calcul sans fin, on arrêtera également le processus, lorsque le nombre d'intervalles considéré est supérieur à un valeur `nmax`.
   Votre fonction retournera trois données : la valeur `I` de l'intégrale calculée, le nombre `n` d'intervalles utilisés pour obtenir l'erreur requise, l'estimation de l'erreur `errorEst`.

[1] On peut évidemment faire appel à la fonction précédente pour cette étape....

3. Comme d'habitude, pour tester votre programme, on vous a fourni un tout petit programme simple `trapezesTest.py` pour intégrer la fonction $\sin(x)$ entre 0 et $\pi/2$. Normalement, vous devriez obtenir une valeur proche de l'unité !

```
ffrom matplotlib import pyplot as plt
from numpy import *

def u(x):
  return cos(x)

a = 0
b = pi/2
n = 10

I = trapezeEasy(u,a,b,n)
errorExact = abs(1.0 - I)
print(" ======= Integral of sinus between 0 and pi/2 = %21.14e " % I)
print("  True error = %14.7e" % errorExact)
print("  Number of intervals = %d" % n)

I,n,errorEst = trapezeFun(u,a,b,n,200000,1e-12)
errorExact = abs(1.0 - I)
print(" ======= Integral of sinus between 0 and pi/2 = %21.14e " % I)
print("  True error = %14.7e" % errorExact)
print("  Est. error = %14.7e" % errorEst)
print("  Number of intervals = %d" % n)

plt.figure("Discovering the trapezoids integration rule :-)")
x = [a,b]
plt.plot(x,u(x),'.k',markersize=5)
x = linspace(a,b,200)
plt.fill(append(x,[0]),append(u(x),[0]),'xkcd:sky blue')
x = linspace(-pi/2,pi,300)
plt.plot(x,u(x),'-k')
plt.title('Integrating sinus between 0 and pi/2')
plt.gca().axhline(y=0,color='k',linewidth=1.0)
plt.show()
```

4. Le devoir est facile et même très très très très facile, mais faut quand-même essayer de réaliser une implémentation la plus efficace possible en évitant les calculs inutiles : donc, c'est facile, mais il y a quand-même une astuce pour les plus futés d'entre vous :-) Ah oui, il faut évidemment pouvoir intégrer autre chose que le sinus au passage :-)

5. Votre fonction (avec les éventuelles sous-fonctions que vous auriez créées) sera soumise sur `zouLab` sans y adjoindre le programme de test fourni ! Cette fonction devra être soumise via le web : ce travail est individuel et sera évalué.