Nom des élèves du groupe :

Activité numérique : Etude de l'évolution d'un système chimique

1. Utiliser un notebook

Dans ce TP, tu vas utiliser un *notebook* Jupyter qui est un document « mixte », contenant du texte et du code Python. Ces lignes, que tu es en train de lire, font partie d'un *notebook* .

Dans la suite de ce *notebook* , on va donc utiliser du code Python. Pas d'inquiétude si tu ne comprends pas toutes les lignes de code. Tu ne vas devoir modifier que certaines lignes. Un *notebook* Jupyter te permet de de modifier et d'exécuter des scripts Python, directement dans le navigateur.

En haut du *notebook* , tu dois voir une barre, contenant :

- un titre pour le *notebook* ;
- une barre de menus avec les entrées `File` , `Insert` , `Cell` , `Kernel` ;
- et une barre de boutons qui sont des raccourcis vers certains menus fréquemment utilisés. Si tu laisses ta souris au dessus d'un bouton, un petit texte apparaît, indiquant à quelle fonction correspond ce bouton.

Un *notebook* est constitué d'une suite de cellules, soit textuelles, soit contenant du code. Les cellules de code sont facilement reconnaissables, elles sont précédées de `Entrée [ ]:` . La cellule qui suit celle que tu es en train de lire est une cellule de code.

Pour commencer, sélectionne la cellule de code ci-dessous avec ta souris et appuie dans la barre de boutons sur celui en forme de flèche triangulaire vers la droite (`Exécuter`).

```
In [11]: 20*30
```

```
Out[11]: 600
```

Comme tu le vois, la cellule est « exécutée » (on dira plus volontiers « évaluée »), et on passe à la cellule suivante.

Alternativement tu peux simplement taper au clavier `SHIFT`+`ENTER`, ou, selon les claviers, `MAJ`+`ENTRÉE`, pour obtenir le même effet. D'une manière générale, il est important d'apprendre et d'utiliser les raccourcis clavier, cela te fera gagner beaucoup de temps par la suite.

La façon habituelle d'exécuter l'ensemble du notebook consiste à partir de la première cellule, et à taper `SHIFT`+`ENTER` (ou `MAJ`+`ENTRÉE`) jusqu'au bout du notebook, en n'allant pas trop vite, c'est-à-dire en attendant le résultat de l'exécution de chaque cellule.

Lorsqu'une cellule de code a été évaluée, Jupyter ajoute sous la cellule `Entrée` une cellule `Out` qui donne le résultat du fragment Python, soit ci-dessus 600.

Jupyter ajoute également un nombre entre les crochets pour afficher, par exemple ci-dessus, `Entrée [1]:` . Ce nombre te permet de retrouver l'ordre dans lequel les cellules ont été évaluées.

Tu peux naturellement modifier ces cellules de code pour faire des essais.

Tu peux également évaluer tout le notebook en une seule fois en utilisant le menu `Cell -> Run All` .

Si tu fais trop de modifications, ou si tu perds le fil de ce que tu as évalué, il peut être utile de redémarrer ton interpréteur. Le menu `Kernel → Restart` te permet de faire cela.

Attention :

Il est important de sauvegarder ton travail très régulièrement. Pour cela il faut cliquer sur le bouton représentant une disquette qui se trouve en haut à gauche dans la barre de boutons (Créer une nouvelle sauvegarde).

2. Travail demandé

Tu devras :

- lire ce *notebook* et exécuter au fur et à mesure toutes les cellules de code (taper au clavier `SHIFT`+`ENTER` ou `MAJ`+`ENTRÉE`).
- analyser les résultats obtenus
- répondre aux questions en sélectionnnant avec ta souris la cellule textuelle située juste en dessous de la question afin de pouvoir y rédiger ta réponse.
- modifier certaines lignes du code quand cela te sera demandé.

3. Problématique

Ce programme permet d'étudier l'évolution des quantités de matière des réactifs et produits d'une réaction dont l'équation est du type : aA + bB --> cC + dD où a, b, c et d sont les nombres stoechiométriques respectifs des espèces chimiques A, B, C et D.

Capacité numérique : Déterminer la composition de l'état final d'un système siège d'une transformation chimique totale à l'aide d'un langage de programmation.

4. Les données d'entrée

Le programme doit tout d'abord connaître le nom des réactifs et des produits puis les valeurs des nombres stoechiométriques et enfin les quantités de matière initiales des réactifs et des produits.

On s'intéresse à la réaction mettant en jeu le diiode I_2 et les ions thiosulfate $S_2O_3^{2-}$ La transformation est modélisée par la réaction d'équation :

$$I_{2\,(aq)} + 2S_2O_{3\,(aq)}^{2-} = 2I_{(aq)}^{-} + S_4O_{6\,(aq)}^{2-}$$

In [12]:
```
# Rattachement des librairies gérant les tracés
# de courbes et certains outils mathématiques
import matplotlib.pyplot as plt
%matplotlib notebook
import numpy as np
```

Saisie du nom des réactifs et des produits

Question 1 : Sur le modèle des lignes de code 2 et 3, modifier les lignes de code nécessaires (lignes 4 et 5) pour entrer le nom des produits C et D.

In [13]:
```
# Nom des réactifs et des produits
nom_A="I2"
nom_B="S2O3--"
nom_C="I-"
nom_D="S4O6--"
```

Saisie des nombres stoechiométriques

Question 2 : Sur le modèle de la ligne de code 2, modifier les lignes de code nécessaires (lignes 3, 4 et 5) pour entrer les valeurs des nombres stoechiométriques b, c et d.

In [14]:
```
# Nombres stoechiométriques
a=1
b=2
c=2
d=1
```

In [15]:
```
# Affichage de l'équation de la réaction
print("L'équation étudiée est : ",
      a," ",nom_A,"    +    ",b," ",nom_B,"    -->     ",
      c," ",nom_C,"      +    ",d," ",nom_D)
```

```
L'équation étudiée est :  1   I2     +     2   S2O3--     -->      2
I-       +     1   S4O6--
```

Saisie des valeurs des quantités de matière initiales

Les valeurs des quantités de matière initiales des réactifs et des produits (exprimées en mole) seront stockées dans des variables notées n_0

(ex : nA_0 pour l'espèce chimique A).

Pour la réaction étudiée on considère que l'on a introduit à l'état initial :
3,5 mmol de diiode I_2 et 5,0 mmol d'ions thiosulfate $S_2O_3^{2-}$

Question 3 : Sur le modèle de la ligne de code 2, ajouter les lignes de code nécessaires (lignes 3, 4 et 5) pour entrer les quantités de matière initiales des autres espèces chimiques en jeu. Attention de bien entrer les valeurs en mol ! Vous pourrez par exemple taper 2.5e-3 pour 2,5 mmol

In [16]:
```
# Quantités de matières initiales en mol
nA_0 = 3.5e-3
nB_0 = 5e-3
nC_0 = 0
nD_0 = 0
```

5. Initialisation des variables

- La variable `Rlimitant` initialement vide aura vocation à accueillir le nom du (ou des) réactif(s) limitant(s) selon la composition du mélange initial.
- L'avancement `x` est initialisé à la valeur 0 mol et un incrément d'avancement `dx` est entré (dx = 0,00001 mol ici).
- Les listes `nA`, `nB`, `nC` et `nD` sont destinées à recueillir les quantités de matière successives des réactifs et des produits au fur et à mesure de l'augmentation de l'avancement.
- La liste `X` stocke les valeurs successives de l'avancement.

In [17]:
```
#Initialisation des variables

# Initialisation de la chaine de caractère correspondant
# au réactif limitant
Rlimitant =''

# Avancement initial
x=0

# Pas d'avancement (on augmentera progressivement x de la valeur dx)
dx=0.00001

# Création des listes contenant les quantités de matière
# et initialisation de ces listes avec la valeur initiale
nA=[nA_0]
nB=[nB_0]
nC=[nC_0]
nD=[nD_0]

# Création et nitialisation de la liste contenant l'avancement
X=[x]
```

6. Calculs des quantités de matière et détermination du réactif limitant

Calculs des quantités de matière en cours d'avancement

Question 4 : Sur le modèle de la ligne 5, modifier les lignes de code 6, 7 et 8 permettant de calculer les quantités de matière du réactif B, ainsi que des produits C et D.

NOTE CODAGE : l'instruction `nA.append(nA_0-a*x)` permet d'ajouter la valeur indiquée entre parenthèses à la fin de la liste nA.

Détermination du réactif limitant

Question 5 : Compléter les tests des lignes de code 11 et 12 en choisissant parmi : `<0` , `<=0` , `>0` et `>=0` .

NOTE CODAGE : L'instruction `nA[-1]` permet d'avoir accès à la dernière valeur de la liste `nA` .

Question 6 : Compléter la ligne de code 13 en choisissant l'opérateur logique adéquat parmi : `and` (ET logique) et `or` (OU logique).

Affichage du nom du réactif limitant et de l'avancement maximal

In [18]:
```
# Calculs des quantités de matière en cours d'avancement
while nA[-1] > 0 and nB[-1] > 0 :
    x=x+dx
    X.append(x)
    nA.append(nA_0-a*x)
    nB.append(nB_0-b*x)
    nC.append(nC_0+c*x)
    nD.append(nD_0+d*x)

#Détermination du réactif limitant
if nA[-1]  <=0                  : Rlimitant = nom_A
if nB[-1]  <=0                  : Rlimitant = nom_B
if nA[-1]  <=0   and   nB[-1]<=0 :
    Rlimitant=nom_A+'  et '+nom_B+' (le mélange est stoechiometrique)'

#Affichage des résultats
print('Réactif limitant :',Rlimitant,
      '\nAvancement maximum : ','{0:.2e}'.format(x),
      'mol' )
# {0:.2e} permet d'afficher un nombre arrondi
# avec 3 chiffres significatifs.
```

```
Réactif limitant : S2O3--
Avancement maximum :  2.50e-03 mol
```

Question 7 : Indiquer la ligne du programme de la cellule précédente qui code l'information correspondant à une transformation totale. Justifier.

Répondre ici.

7. Affichage des courbes permettant de suivre l'évolution des quantités de matière

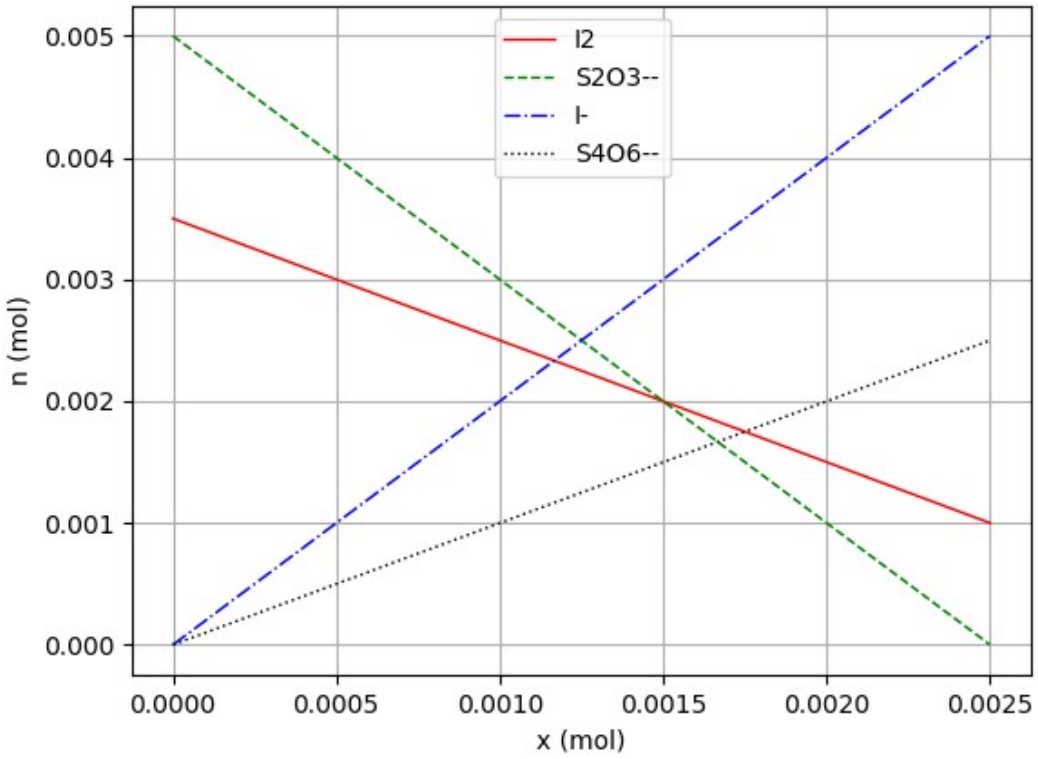
Les lignes de code 2, 3 et 4 ci-dessous permettent d'afficher les courbes de l'évolution des quantités de matière de A, B et C en fonction de l'avancement x.

Question 8 : Compléter la ligne 5 pour afficher la courbe correspondant à l'évolution de la quantité de matière de l'espèce chimique D en fonction de l'avancement. Cette courbe devra être noire avec une ligne en pointillé court et de la même largeur que les trois autres courbes.

NOTE CODAGE : la commande `plt.plot` peut être enrichie de divers arguments (comme ici avec `'r-'` = r pour red et - pour ligne en trait plein) :

- Couleur : `r` (red), `k` (black), `b` (blue), `y` (yellow), `g` (green)
- Style de ligne : `-` (ligne en trait plein), `--` (ligne en pointillé long), `-.` (ligne en pointillé mixte), `:` (ligne en pointillé court)
- `lw` signifie linewidth (largeur de la ligne)
- `label` permet de légender la courbe en créant une étiquette

In [19]:
```
# Affichage des courbes permettant de suivre l'évolution des quantités de matière
plt.plot(X,nA,'r-',lw=1,label=nom_A)
plt.plot(X,nB,'g--',lw=1,label=nom_B)
plt.plot(X,nC,'b-.',lw=1,label=nom_C)
plt.plot(X,nD,'k:',lw=1,label=nom_D)
plt.grid()
plt.xlabel('x (mol)')
plt.ylabel('n (mol)')
plt.legend()
plt.show()
```



Question 9 : Sans regarder la légende, comment peut-on distinguer graphiquement les courbes des réactifs des courbes des produits ?

Répondre ici.

Question 10 : Déterminer graphiquement les quantités de matière des différentes espèces à l'état final.

Répondre ici.

Question 11 : Quel est le réactif en excès ? et comment modifier sa quantité initiale pour obtenir un mélange stoechiométrique ?

Répondre ici.

8. Modélisation des droites obtenues

Les lignes de code suivantes vont permettre de modéliser chacune des 4 droites obtenues sur le graphe ci-dessus. Ces droites sont de type linéaire ou affine et peuvent être modélisées avec un polynôme de degré 1 : ax+b (où x est à la puissance 1). Les résultats des quatre modélisations sont ensuite affichés pour analyse.

NOTE CODAGE :
L'instruction `p=np.polyfit(x,y,deg)` permet de modéliser une courbe `y` =f(`x`) avec un polynome de degré `deg` . La fonction polyfit de la bibliothèque numpy retourne un tableau numpy `p` à une dimension contenant les coefficient du polynôme :

$$P(x) = p[0] \times x^{deg} + p[1] \times x^{deg-1} \ldots + p[deg]$$

Pour un polynôme de degré 1 :

$$P(x) = p[0] \times x + p[1]$$

Question 12 : Compléter les lignes de code 2, 3 et 4 (sur le modèle de la ligne 1) pour modéliser les courbes concernant l'évolution des quantités de matière des espèces chimiques B, C et D.

In [20]:
```
Amodel=np.polyfit(X, nA,1)
Bmodel=np.polyfit(X, nB,1)
Cmodel=np.polyfit(X, nC,1)
Dmodel=np.polyfit(X, nD,1)

print ("La droite représentant l'évolution de la quantité n de",
       '{:8}'.format(nom_A),
       " en fonction de x a pour équation : n = ",
       '{0:.0f}'.format(Amodel[0]),'{:5}'.format("x +"),
       "{0:.2e}".format(Amodel[1]) )

print ("La droite représentant l'évolution de la quantité n de",
       '{:8}'.format(nom_B),
       " en fonction de x a pour équation : n = ",
       '{0:.0f}'.format(Bmodel[0]),'{:5}'.format("x +"),
       "{0:.2e}".format(Bmodel[1]) )

print ("La droite représentant l'évolution de la quantité n de",
       '{:8}'.format(nom_C),
       " en fonction de x a pour équation : n =  ",
       '{0:.0f}'.format(Cmodel[0]),'{:5}'.format("x +"),
       "{0:.2e}".format(Cmodel[1]) )

print ("La droite représentant l'évolution de la quantité n de",
       '{:8}'.format(nom_D),
       " en fonction de x a pour équation : n =  ",
       '{0:.0f}'.format(Dmodel[0]),'{:5}'.format("x +"),
       "{0:.2e}".format(Dmodel[1]) )
```

```
La droite représentant l'évolution de la quantité n de I2        en f
onction de x a pour équation : n =  -1 x +   3.50e-03
La droite représentant l'évolution de la quantité n de S2O3--    en f
onction de x a pour équation : n =  -2 x +   5.00e-03
La droite représentant l'évolution de la quantité n de I-        en f
onction de x a pour équation : n =   2 x +   0.00e+00
La droite représentant l'évolution de la quantité n de S4O6--    en f
onction de x a pour équation : n =   1 x +   0.00e+00
```

Question 13: En quoi les équations des courbes modélisées sont-elles cohérentes avec les données du problème ?

Répondre ici.

Enregistrer ce notebook au format html

Une fois que vous avez terminé de répondre aux questions, enregistrez ce notebook au format html en utilisant le menu `File` -> `Download as` -> `HTML (.html)` et envoyez le moi par la messagerie Pronote. Vérifiez que vous avez bien indiqué en haut du fichier le nom de tous les élèves du groupe.

In [ ]: