éduSCOL
Informer et accompagner les professionnels de l'éducation

Liberté • Égalité • Fraternité
RÉPUBLIQUE FRANÇAISE

MINISTÈRE DE L'ÉDUCATION NATIONALE ET DE LA JEUNESSE

VOIE GÉNÉRALE

2DE | 1RE | TLE

Physique-chimie

ENSEIGNEMENT SPÉCIALITÉ

PROGRAMMER EN PYTHON
FICHE N°1 : RÉACTIF LIMITANT, BOUCLE WHILE ET FONCTION

Le langage Python est ici utilisé pour déterminer l'avancement maximal d'une transformation et identifier le réactif limitant à partir de la donnée des quantités de matière initiales pour une réaction donnée. D'un point de vue algorithmique, c'est l'occasion d'utiliser une boucle **while** et ainsi d'appréhender la gestion d'une condition d'arrêt qui n'est pas connue a priori.

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.

Premier exemple, fonction input, première boucle

La fiche s'appuie sur la réaction mettant en jeu le diiode et les ions thiosulfate en solution aqueuse :

$$I_2(aq) + 2\,S_2O_3^{2-}(aq) = 2\,I^-(aq) + S_4O_6^{2-}(aq)$$

Le script débute par la saisie des quantités de matière initialement introduites des deux réactifs (nI2_0 et nS2O3_0). Pour rendre le script interactif, ces quantités sont entrées grâce à la commande input qui permet de poser une question à l'utilisateur. La réponse à la question étant par défaut interprétée par python comme une chaîne de caractère (en quelque sorte, un texte), l'instruction float convertit celle-ci en un nombre.

```
I2_0 = input('quantité initiale en diiode en mol :')
nI2_0 = float(I2_0)
S2O3_0 = input('quantité initiale en thiosulfate en mol :')
nS2O3_0 = float(S2O3_0)
```

L'avancement est initialisé à la valeur x = 0 mol, et un incrément d'avancement est entré (a = 0,001 mol ici).

Une variable 'limitant', initialement vide, aura vocation à accueillir le nom du (ou des) réactifs limitants selon la composition du mélange initial.

Les listes qI2 et qS2O3 sont destinées à recueillir les quantités de matière successives de ces deux réactifs.

```
limitant ='' # initiallisation de la chaine de caractère correspondant au réactif limitant
x=0 # avancement initial
a=0.001 #pas d'avancement
qI2=[nI2_0]
qS2O3=[nS2O3_0]
```

Le script opère ensuite par augmentation progressive de l'avancement tant que les deux réactifs sont présents.

La boucle conditionnelle while permet d'indiquer à Python qu'il doit poursuivre l'augmentation de l'avancement tant que les quantités de matière des deux réactifs sont positives.

La fonction append sert à ajouter aux listes créées chaque nouvelle valeur des quantités de matière des réactifs.

```
while qI2[-1]>0 and qS2O3[-1]>0:
    x=x+a
    qI2.append(nI2_0-x)
    qS2O3.append(nS2O3_0-2*x)
```

Quand la transformation, totale ici, a été conduite à son terme, le(s) réactif(s) limitant(s) est identifié grâce à la commande liste[-1] qui permet d'appeler la dernière valeur dans une liste.

L'avancement final est choisi comme la dernière valeur d'avancement calculée avant disparition d'un réactif.

Enfin, la commande print permet d'afficher l'état final. La commande round permet d'afficher un arrondi à 2 chiffres significatifs.

```
#résolution du problème et affichage du résultat
if qI2[-1]<=0:
    limitant = 'diiode'
if qS2O3[-1]<=0:
    limitant = 'thiosulfate'
#print(limitant)
print('Le réactif limitant est le ',limitant,'\n Avancement maximum : ',round(x,2),'mol' )
```

Dans les conditions initiales choisies, le script renvoie :

```
quantité initiale en diiode en mol :2
quantité initiale en thiosulfate en mol :3
 Le réactif limitant est le  thiosulfate
  Avancement maximum :  1.5 mol
```

Avec les élèves, il est possible de tester différentes situations initiales pour mettre l'accent sur l'importance des nombres stœchiométriques dans l'identification du réactif limitant (le réactif limitant n'est pas nécessairement celui introduit en plus faible quantité).

Généralisation, première définition, deuxième boucle

Ce script peut être généralisé afin de gérer tout cas de nombres stœchiométriques et de conditions initiales, pour un système siège d'une réaction de la forme :

$$a_A\ A + a_B\ B = a_C\ C + a_D\ D$$

Sous Python, la définition d'une fonction débute par « def nom_procedure(arguments) : » et termine par return.

```
def react_lim(aA,aB,nA,nB) :
    x=0             # Initialisation de l'avancement
    dx=0.00001      # Incrément d'avancement
    qA=[nA]         # Liste stockant les quantités de matière successives de A
    qB=[nB]         # Idem pour B
    RL=[]           # Liste qui stockera le nom du réactif limitant
    while qA[-1]>0 and qB[-1]>0 :
        x=x+dx
        qA.append(nA-aA*x)
        qB.append(nB-aB*x)
    if qA[-1]<=0 :
        RL.append('A')
    if qB[-1]<=0 :
        RL.append('B')
    return(RL,round(x,2))

react_lim(aA,aB,nA,nB)
```

Dans le cas d'une transformation mettant en jeu initialement 2 mol d'un réactif « A » et 3 mol d'un réactif « B », modélisée par une réaction de stœchiométrie « 5 A pour 2 B », la commande react_lim(5,2,2,3) renvoie le résultat :

```
Out[4]:  (['A'], 0.4)
```

Ce résultat peut s'interpréter ainsi : « le réactif limitant est A, la valeur maximale de l'avancement est 0,4 mol ».

Ici encore, la simulation de compositions initiales différentes est l'occasion de faire remarquer aux élèves que le réactif limitant n'est pas forcément celui qui est introduit en plus faible quantité.

La définition d'une procédure généralisée de ce type ne présente vraiment d'intérêt que si elle est insérée dans un programme plus complet, visant par exemple, à déterminer de manière automatisée le réactif limitant et la valeur maximale de l'avancement dans un grand nombre de situations, comme c'est le cas pour la simulation d'un titrage.