

Code de partage avec Cappytale : 23a1-1072903

[Corrigé](#)

## 1 Echauffement

Lors du prochain devoir, je vous propose de partager 400 points de manière équitable et d'attribuer une note entière à chacun.

Combien restera-t-il de points après partage ?

$800 = 17 \times 46 + 18$ , donc il restera 18 points après partage.

## 2 Exercices

### Exercice 1 - division euclidienne d'entiers positifs

Ecrire une fonction qui prend deux entiers  $n$  et  $p$  comme arguments et qui renvoie le quotient et reste de la division euclidienne de  $n$  par  $p$ .

De manière concrète cela revient à effectuer un partage équitable d'un « tas » de  $n$  unités à  $p$  personnes. On peut réaliser ce partage en distribuant une unité à chacune des  $p$  personnes, puis à réitérer l'opération (une tournée) tant que le tas restant est suffisant. Ce que l'on simule par la fonction ci-contre, que l'on peut tester avec l'exemple précédent : `div_eucl(800,46)` qui doit renvoyer (17,18)

```
def div_eucl(n,p):
    tas=n
    tournee=0
    while tas>=p:
        tas=tas-p
        tournee=tournee+1
    return tournee,tas
```

### Exercice 2 - division euclidienne de polynômes - un cas particulier

On rappelle que comme pour les entiers, on peut effectuer des divisions avec les polynômes, le reste (un polynôme, noté  $R$  ici) étant de degré strictement inférieur au diviseur (un autre polynôme, noté  $B$  ici) dans ce cas. Et on parlera aussi de quotient :  $A = BQ + R$

Cas d'une division par un polynôme de degré 1 :

1. Ecrire une fonction qui demande à l'utilisateur de rentrer un polynôme  $P$  et qui renvoie le terme de plus haut degré du polynôme  $Q$  dans la division euclidienne de  $P(x)$  par  $x - 2$ . Plus précisément si  $Q(x) = 1 - 7x + 3x^3$ , alors on souhaite que la fonction renvoie `[0,0,0,3]` (qui correspond à  $3x^3$ )

*on pourra utiliser la fonction `degre` définie à la séance précédente.*

En développant et en identifiant, on remarque que le coefficient dominant de  $Q$  sera le même que celui de  $P$  (attention leurs degrés sont différents par contre). Par exemple si  $P(x) = 4x^4 + \dots$  alors  $Q(x) = 4x^3 + \dots$

Il suffit donc de renvoyer un « polynôme » (une liste), dont la taille vaut un de moins que celle de  $P$  et dont tous les coefficients valent 0 sauf le dernier qui vaut le coefficient dominant de  $P$

```
def terme(P):
    n=degre(P)
    Q=[0 for n in range(0,n)]
    Q[n-1]=P[n]
    return Q
```

On peut le tester par exemple avec `terme([0,1, 2, 3])`

2. A l'aide de l'algorithme de la division euclidienne, écrire une fonction qui effectue la division euclidienne de  $P(x)$  par  $x - 2$  et donc qui renvoie le quotient (le polynôme  $Q$  sous la forme d'une liste de nombres) et le nombre  $r$  de l'égalité suivante :

$$P(x) = (x - a)Q(x) + r$$

L'algorithme consiste à retirer de  $P$  le produit de  $x - 2$  et du polynôme « terme » calculé plus haut. Le calcul du polynôme « terme » nous permet de déterminer le coefficient dominant de  $Q$  et la soustraction permet de créer un polynôme  $P2$  dont le degré vaudra une unité de moins que celui  $P$ , on applique alors la même méthode à  $P2$  et  $Q2$  (le polynôme  $Q$  moins son terme de plus haut degré) et on trouve le deuxième coefficient de plus haut degré de  $Q$  et ainsi de suite. La condition d'arrêt porte sur le degré : on continue tant que ce qu'il reste (dans le polynôme  $P_n$  est de degré plus grand que le diviseur (ici 1 car le diviseur est  $x - 2$ ))

```
def division(P) :
    n=degre(P)
    Q=[0 for k in range(0,n)]
    P1=P
    while n>=1:
        A=terme(P1) # on calcule le terme de plus haut degré de Q
        B=[0 for k in range(0,n+1)] # on crée le polynôme B, qui est A*(x
-2)
        #B[n]= A[n-1] #optionnel en fait
        B[n-1]= -2*A[n-1]
        #il n'y a que 2 coefficients à changer dans P

        P1[n]=0
        P1[n-1]=P1[n-1]-B[n-1]
        Q[n-1]=A[n-1]
        n=degre(P1)# on met à jour le degré de P
        # normalement, le P[n] vaut alors 0, ce qui abaisse le degré et
permet d'avancer dans la boucle
        R=[P1[0]]
    return [Q,R]
```