TD 5 :
Arithmétique

MPSI - Lycée Thiers

Exercice 1

Exercice 1. Ecrire une fonction `pgcd(a,b)` qui calcule le PGCD de deux entiers positifs a et b à l'aide de l'**algorithme d'Euclide** :

```
Donnés deux entiers positifs a et b :
TANT QUE b ≠ 0 :
    Changer a et b respectivement par b et a modulo b
RETOURNER a
```

En déduire une fonction `ppcm(a,b)` qui retourne le PPCM (plus petit commun multiple) de deux entiers positifs a et b.

Exercice 1 : Corrigé

```
def pgcd(a,b):
    while b != 0:
        a, b = b, a%b
    return a
```

- On en déduit la fonction `ppcm(a,b)` grâce à la propriété :

$$pgcd(a,b) \times ppcm(a,b) = a \times b$$

```
def ppcm(a,b):
    return a*b // pgcd(a,b)
```

Exercice 2 : Test de primalité

Exercice 2. Ecrire une fonction `premier(n)` prenant en paramètre un entier positif `n` et qui renvoie le booléen `True` ou `False` selon si `n` est un nombre premier ou pas. (Indication : Pour déterminer si un nombre n est premier il suffit de vérifier qu'il n'est divisible par aucun entier compris entre 2 et $\sqrt{n}$.)

Exercice 2 : Correction

```
def premier(n):
    if n==0 or n==1:
        return False
    for i in range(2,int(n**0.5)+1):
        if n%i == 0:
            return False
    return True
```

- Pour obtenir la liste des premiers inférieurs à n :

```
def Premiers(n):
    L = []
    for k in range(2,n+1):
        if premier(k):
            L.append(k)
    return L
```

Exercice 3 : Décomposition en facteurs premiers

Exercice 3. Ecrire une fonction `decomposition(n)` prenant en paramètre un entier `n` positif et qui retourne sous forme d'une liste sa décomposition en facteurs premiers. Pour donner la décomposition d'un nombre n en facteurs premiers, on peut utiliser l'algorithme suivant :

```
p = 2
TANT QUE n ≠ 1:
    SI p divise n:
        Ajouter p à la liste des diviseurs premiers de n
        Changer n en n/p
    SINON:
        Changer p en p+1
RETOURNER la liste des diviseurs premiers de n
```

Exercice 3 : Corrigé

```
def decomposition(n):
    L = []
    p = 2
    while n!=1:
        if n%p == 0:
            L.append(p)
            n = n//p
        else:
            p+=1
    return L
```

Exercice 4 : programme d'arithmétique

Exercice 4. Ecrire un programme qui affiche le menu suivant, attend le choix de l'utilisateur puis effectue les actions correspondantes :

```
 MENU :
1 - Calculer le pgcd de 2 entiers
2 - Calculer le ppcm de 2 entiers
3 - Déterminer si un nombre est premier
4 - Donner la décomposition en facteurs premiers d'un nombre
q - Quitter
```

Exercice 4 : Corrigé

- L'écriture du menu s'obtient par :

```
while True:
    print("""MENU:
    1 - calculer le pgcd de deux nombres
    2 - calculer le ppcm de deux nombres
    3 - Déterminer si un nombre est premier
    4 - Donner la decomposition en facteurs premiers d'un nombre
    q = Quitter""")
    rep = input("Votre choix :")
```

Une chaîne de caractère placée entre 3 guillemets """.""" permet saut à la ligne et tabulation.

Exercice 4 : Corrigé

Pour sortir de la boucle : `break`.

```
    if rep == "q":
        break
```

Ensuite :

```
    elif rep == '1':
        a = int(input('Saisir un entier: '))
        b = int(input('Saisir un entier: '))
        print("Leur PGCD est",pgcd(a,b))
        input()
    elif rep == '2':
        a = int(input('Saisir un entier: '))
        b = int(input('Saisir un entier: '))
        print("Leur PPCM est",ppcm(a,b))
        input()
```

Exercice 4 : Corrigé

```
elif rep == '3':
    n = int(input('Saisir un entier: '))
    if premier(n):
        print(n,"est premier")
    else:
        print(n,"n'est pas premier")
    input()
elif rep == '4':
    n = int(input('Saisir un entier: '))
    L = decomposition(n)
    print(n," = ",end="",sep="")
    for x in L:
        print(x,'.',end="",sep="")
    input()
```

Les options sep et end de la fonction print définissent ce qu'écrit print pour chaque virgule et en fin de commande.