Exercice II.1 *(Variables, conditions et fonctions)*

Pour chacun des programmes suivants, donnez les valeurs de chaque variable à la fin de l'éxécution du programme, et indiquez ce qui s'affiche.

```
x,y=3.5,5
z=x/y
p=7

if (x>3)and(z*10!=p):
    print("Blue")
elif (p>y)or(x>4):
    print("Red")
else:
    print("Black")
```

```
i,j,k=1,2,3
x,y,z=1.1,1.2,1.3
i=i+x
j=j+y
k=k+z
print(i,j,k,x,y,z)
if (i+j>k)and(j-i==x):
    print("Blue")
else:
    print("Red")
```

```
def f1(x,y):
    if (x<y):
        return 3
    else:
        return 4

def f2(n):
    return 2*n-1

x=3.4
y=5.3
print(f2(f1(y,x)))
```

Exercice II.2 *(Variables locales et globales)*
Pour chacun des programmes suivants, indiquez quelles variables sont locales, et lesquelles sont globales. Donnez les valeurs de chaque variable à la fin de l'éxécution du programme, et indiquez ce qui s'affiche.

```
p=4

def mult(x):
    res = p*x
    return res

print(p)
print(mult(3))
print(p)
```

```
p=4

def mult(x):
    p=5
    res = p*x
    return res

print(p)
print(mult(3))
print(p)
```

```
p=4

def mult(x):
    global p
    p=5
    res = p*x
    return res

print(p)
print(mult(3))
print(p)
```

Le fonctionnement des variables locales et globales peut être résumé ainsi :
— les arguments des fonctions sont des variables locales
— par défaut, les fonctions ont accès aux variables globales uniquement pour les lire, mais pas pour les modifier
— si on souhaite permettre à une fonction de modifier une variable globale `p`, il faut utiliser la déclaration `global p`

Exercice II.3 *(Diviseurs d'un nombre)*

1. Écrivez un *programme* Python qui demande à l'utilisateur de saisir un entier naturel n et affiche tous ses diviseurs.
2. Un nombre est *parfait* si la somme de ses diviseurs est égale à deux fois ce nombre, *abondant* si cette somme est strictement supérieure à deux fois ce nombre, *déficient* si la somme de ses diviseurs est strictement inférieure à deux fois ce nombre. Par exemple, 6 est parfait, 12 est abondant, 5 est déficient. Écrivez une *fonction* qui prend en entrée un entier naturel n et renvoie 0 si n est parfait, 1 si n est abondant, et -1 si n est déficient.

3. Ecrivez un *programme* Python qui demande à l'utilisateur de saisir un entier naturel et affiche un message indiquant si l'entier naturel saisi est parfait, abondant ou déficient. Votre programme utilisera la fonction définie à la question précédente.

Exercice II.4 *(Minimum de fonction)*

1. On considère le polynôme P défini par $P = X^3 - 10\ X^2 - 2610\ X + 84$. Ecrivez une *fonction* qui, étant donné un entier naturel x, retourne la valeur de ce polynôme en x.
2. Ecrivez un *programme* qui affiche la plus petite valeur prise par ce polynôme pour les entiers appartenant à l'intervalle $[-50, 50]$.
3. Ecrivez un *programme* qui affiche un entier naturel appartenant à l'intervalle $[-50, 50]$ pour lequel ce minimum est atteint.

Exercice II.5 *(Suite de Syracuse)*
La suite de Syracuse d'un entier N est définie par $u_0 = N$ et pour tout entier naturel $n \geqslant 0$, $u_{n+1} = 3u_n + 1$ si u_n est impair et $u_{n+1} = u_n/2$ si u_n est pair. On conjecture que pour tout entier N, il existe un terme de sa suite de Syracuse égal à 1.

1. Écrivez un programme qui demande un entier N à l'utilisateur et affiche tous les termes de la suite jusqu'au premier terme égal à 1.
2. Écrivez un programme qui, pour tout entier N, calcule le plus petit indice $i(N)$ pour lequel $u_{i(N)} = 1$.
3. Écrivez un programme qui calcule les 100 premiers termes de la suite $(i(N))_{N \geqslant 0}$.
4. Écrivez un programme qui calcule l'entier $N \leqslant 100$ pour lequel $i(N)$ est maximal.
5. Écrivez un programme qui, pour tout entier N, calcule la plus grande valeur prise par les termes de la suite de Syracuse de N.
6. Écrivez un programme qui trouve l'entier $N \leqslant 100$ pour lequel la plus grande valeur prise par les termes de la suite de Syracuse de N est maximale.

Exercice II.6 *(Problèmes algorithmiques)*

Proposez des fonctions Python réalisant les tâches suivantes. On suppose que l'on dispose de la fonction Premier.

1. Etant donné un entier naturel n, calculer le n-ème nombre premier.
2. Etant donnés deux nombres m et n, calculer le plus petit entier naturel supérieur ou égal à 2 qui divise ces deux nombres. On retournera 1 si un tel nombre n'existe pas.
3. Etant donnés deux nombres m et n, calculer le plus grand diviseur commun de m et n.
4. Etant donné un entier naturel n, afficher la décomposition en produit de nombres premiers de n.