Exercice I.1 *(Boucle (2 points))*
Ecrire un *programme* demandant à l'utilisateur de saisir un entier naturel n et affichant tous les entiers i compris entre 1 et n tels que i divise n et i est un multiple de 3.

Par exemple, pour $n = 12$, votre programme affichera les valeurs : 3, 6, 12.

Exercice I.2 *(Manipulation de variables (2 points))*

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

```
# Programme 1

i,j,k,l=3,1,30,10
j=k/l
j=j*j
i=(l*i)-k
if (i==j) or (j!=0 and k>l):
   print("Blue")
else:
   print("Red")
```

```
# Programme 2

y,z=2,3
j=y**z
i=y*z

if (i!=j) and (2*i!=j):
   print("Blue")
else:
   print("Red")
```

Exercice I.3 *(Variables et fonctions (2 points))*
On considère le programme suivant, expliquez précisément son comportement en donnant les valeurs des arguments des fonctions, et les valeurs de retour de ces fonctions, au cours de l'exécution du programme.

```
# Programme 3

def truc(x):
  j=99
  p=j*x
  return p
```

```
def chose(j):
  x=(j+1)%100
  return x

x=100
p=truc(chose(x))
print(p)
```

Exercice I.4 *(Intégrale de Riemann (3+3+1+2 points))*
Cet exercice a pour but de proposer un algorithme pour le calcul de l'intégrale d'une fonction, fondé sur l'intégrale de Riemann.

1. On considère la fonction $f : \mathbb{R} \to \mathbb{R}$ définie par :

$$f : x \mapsto \begin{cases} 2 * x^2 + 6 * x + 5 & \text{si } x < -2 \\ x^2 - 7 * x - 17 & \text{si } -2 \leqslant x < 7 \\ -x^2 + 4 * x + 4 & \text{si } x \geqslant 7 \end{cases}$$

   On peut facilement vérifier que la fonction f est continue sur $\mathbb{R}$.
   Ecrire une *fonction* nommée `fonctionf` ayant comme argument une variable x et retournant la valeur de $f(x)$. Elle sera donc déclarée à l'aide de la ligne suivante :
   `def fonctionf(x):`

2. On rappelle la définition de l'intégrale de Riemann d'une fonction f sur un intervalle $[a, b]$, valable si f est continue sur l'intervalle $[a, b]$:

$$\int_a^b f(x)dx = \lim_{n \to +\infty} u_n \qquad \text{où } u_n = \frac{b-a}{n} * \sum_{i=0}^{n-1} f(a + i * \frac{b-a}{n})$$

   Ecrire une *fonction* nommée `somme` qui a en argument trois variables a, b et n, et retourne la valeur de $\sum_{i=0}^{n-1} f(a + i * \frac{b-a}{n})$. Elle sera déclarée par :
   `def somme(a,b,n):`

3. Ecrire une *fonction* nommée `integrale` qui a en argument trois variables a, b et n, et retourne la valeur du terme u_n. Cette fonction utilisera la fonction `somme` définie précédemment. Elle sera déclarée par :
   `def integrale(a,b,n):`

4. Ecrire un *programme* demandant à l'utilisateur de choisir un entier naturel n et affichant l'approximation de l'intégrale de la fonction f calculée entre $a = -10$ et $b = 10$, donnée par le terme u_n. Si la valeur obtenue est par exemple 36.7, votre programme affichera le message suivant :
   `La valeur approchée calculée est : 36.7`

Exercice I.5 *(Minima (3+2 points))*

On suppose qu'une fonction `polynome` est définie dans le programme python dans lequel vous intervenez. Cette fonction prend en entrée une variable et retourne un entier relatif. Dans cette exercice, vous pouvez donc utiliser la fonction `polynome` en supposant qu'elle est définie.

1. Ecrire une *fonction* `minimum` qui prend en entrée un entier naturel n, et retourne la valeur minimale prise par la fonction `polynome` sur l'ensemble $\{0, 1, \ldots, n\}$. Elle sera déclarée par la ligne suivante :
   `def minimum(n):`

2. Ecrire une *fonction* `compte` qui prend en entrée un entier naturel n et un entier (relatif) x, et retourne le nombre d'éléments $i \in \{0, 1, \ldots, n\}$ pour lesquels la fonction `polynome` prend la valeur x. Elle sera déclarée par la ligne suivante :
   `def compte(n,x):`