## 5. Complément sur les variants et invariants de boucle

Un programme peut être amené à manipuler un grand nombre de données dont les valeurs peuvent évoluer au cours de son exécution. Il est important de s'assurer que le programme réalise bien le traitement voulu et qu'il s'arrêtera en retournant le bon résultat. C'est par la correction du programme que l'on pourra s'en assurer.
Cette correction consiste à trouver :

- Un **variant de boucle** : condition de sortie d'une boucle.
- Un **invariant de boucle** : propriété logique qui reste vraie de l'entrée de la boucle jusqu'à sa sortie.

Le **variant de boucle** est la variable qui croit ou décroit à chaque itération de la boucle. Cette variable comparée à une valeur constante doit permettre de mettre fin à la boucle après un nombre fini d'itérations.

```
# exemple 1
for i in range(10):
    # faire quelque chose

# exemple 2
i = 0
while i < 10:
    # faire quelque chose
    i=i+1
```

### 5.1. Rappel sur la notion d'invariant de boucle

Un invariant est une hypothèse pouvant être vraie ou fausse. Les **invariants de boucles** servent à prouver la validité d'un algorithme comportant une ou plusieurs boucles for ou while. Un invariant de boucle est une proposition qui :

- est vraie avant d'entrer dans la boucle (initialisation)
- reste vraie après une itération si elle était vraie avant (conservation)
- donne le résultat attendu en fin de boucle (terminaison)

Si pour un algorithme donné, il existe un invariant qui vérifie ces trois étapes, alors on dit que l'algorithme est valide.

EXERCICE 3 : On considère la fonction suivante qui permet de calculer le quotient et le reste de la division euclidienne d'un entier positif par un entier strictement positif :

```
Fonction: division(a, b)
Action: Calcul du quotient q et du reste r de la division euclidienne de l'entier a par l'entier b
Début
        Si a<0 ET b<0 Alors
                Retourner Aucun
        q ← 0
        r ← a
        TantQue r >= b Faire
                q ← q + 1
                r ← r – b
        FinTantQue
        Renvoyer q,r
Fin
```

1- Décrire le fonctionnement de l'algorithme pour les arguments (a = 17, b = 5) au moyen d'un tableau indiquant l'évolution des valeurs des variables au fil des itérations.

| Itération | a | b | r | q | Condition r >= b |
|---|---|---|---|---|---|
| | 17 | 5 | 17 | 0 | |
| Itération 0 | 17 | 5 | | | |
| Itération 1 | 17 | 5 | | | |
| Itération 2 | 17 | 5 | | | |
| Itération 3 | 17 | 5 | | | |

2- Montrer que la boucle TantQue se termine en utilisant un variant de boucle.

3- Montrer que la propriété a = b*q+r est un invariant de la boucle TantQue, en déduire que l'algorithme produit le résultat attendu.

Initialisation :

Conservation : pour chaque itération b*q+r

| Itération | Valeur de q | Valeur de r | Valeur de b*q+r |
|---|---|---|---|
| i | x | y | bx+y |
| i+1 | | | |

Terminaison :

4- Saisir en langage python le script de la division euclidienne en utilisant la fonction division_entiere() prenant comme paramètre deux entiers positifs a et b.

5- Rajouter un assert pour contrôlant l'invariant de boucle à l'initialisation.

6- Rajouter un assert pour contrôlant l'invariant de boucle à la conservation.

7- Rajouter un assert pour contrôlant l'invariant de boucle à la terminaison. Pour cela vous ferrez appel à une fonction de validation préalablement crée. Cette fonction de validation () prendra comme paramètre a et b et renverra q et r.