Énoncé

1. Branchements

On rappelle qu'une proposition est un expression formée avec des noms de l'espace courant (variables) et qui s'évalue à une valeur booléenne.

Algorithme 1 : Enchaînement de branchements conditionnels

```
if a > b then
    t ← a
    a ← b
    b ← t
#pause 1
if a > c then
    t ← a
    a ← c
    c ← t
#pause 2
if b > c then
    t ← b
    b ← c
    c ← t
#pause 3
#fin
```

Pour l'algorithme 1, former trois propositions $\mathcal{P}_1$, $\mathcal{P}_2$, $\mathcal{P}_3$ telle que $\mathcal{P}_i$ s'évalue à `vrai` à la pause i. Justifier que les trois propriétés s'évaluent encore à `vrai` à la fin. Que peut-on en conclure ?

2. Méthode de Hörner

Pour la boucle de l'algorithme 2, former un variant et montrer que

$$\Phi = \left(y == a_i + a_{i+1}x + \cdots + a_n x^{n-i}\right)$$

est un invariant de boucle. Que désigne y à la fin ?

Algorithme 2 : Méthode de Hörner

```
A ← liste de n + 1 nombres indexée à partir de 0.
#on note a_i le nombre désigné par A[i].
y ← A[n]
i ← n
tant que i ≥ 1
    i ← i − 1
    y ← A[i] + x ∗ y
```

3. Tri bulle

Une *montée* dans une liste A dont les valeurs sont des objets comparables est un couple d'indices (i, j) tels que $A[i] < A[j]$ avec $i < j$.

Algorithme 3 : tri bulle

```
A ← une liste de n objets comparables indexée à partir de 0.
i ← 0
tant que i < n-1
    if A[i] < A[i+1] then
        permuter les valeurs d'indice i et i + 1 dans A.
        if i > 0 then
            i ← i − 1
    else
        i ← i + 1
```

Pour la boucle de l'algorithme 3 , $n - i$ est-il un variant ? Le nombre de montées est-il un variant ? Former un variant et un invariant. En les utilisant, prouver que la boucle termine et, qu'après la sortie, les valeurs de A sont décroissantes avec les indices.
Implémenter l'algorithme en Python dans une fonction `TriBulle` qui prend comme seul paramètre la liste à ranger et qui renvoie le nombre de permutations effectuées. Après l'exécution d'une ligne `print(TriBulle(A))`. Que désigne `A` ?

4. Développement de Engel

Soit $b \in \mathbb{Z}$ et $a \in \mathbb{N}^*$, la division euclidienne de b par a se traduit par :

$$\exists q \in \mathbb{Z} \text{ et } r \in [\![0, a[\![\text{ tq } b = qa + r$$

On désigne par *pseudo division euclidienne* (pde) de b par a la propriété :

$$\exists q \in \mathbb{Z} \text{ et } r \in [\![0, a[\![\text{ tq } b = qa - r$$

On dira encore que q est le quotient et r le reste dans la pseudo division de b par a.

1. Exprimer avec les symboles $\lfloor \ \rfloor$ et $\lceil \ \rceil$ le quotient de b par a pour chaque division.

Algorithme 4 : Un développement

$b \leftarrow$ un naturel non nul
$a \leftarrow a_{ini}$ un naturel non nul strictement plus petit que b
$L \leftarrow$ une liste vide
$s \leftarrow 0$
$Q \leftarrow 1$
tant que $a > 0$
  $q \leftarrow$ quotient de la pde de b par a
  $r \leftarrow$ reste de la pde de b par a
  $a \leftarrow r$
  $Q \leftarrow Q * q$
  $s \leftarrow s + \frac{1}{Q}$
  placer Q à la fin de L

---

2. Pour la boucle de l'algorithme 4, montrer que a est un variant et

$$\Phi = \left(sb + \frac{a}{Q} == a_{ini} \right)$$

est un invariant. Que désigne a après la sortie de la boucle ? Quelle sorte de décomposition cet algorithme permet-il de calculer ?

Corrigé

5. Branchements

On peut choisir

$$\mathcal{P}_1 = (a \leq b) \qquad \mathcal{P}_2 = (a \leq c) \qquad \mathcal{P}_3 = (b \leq c)$$

Lors du deuxième branchement, la valeur de a diminue ou ne change pas, celle de b ne change pas, la condition $\mathcal{P}_1$ reste donc vraie. La valeur de a est donc inférieure ou égale aux deux autres valeurs
Lors du troisième branchement, a ne change pas et les valeurs de b et c sont conservées ou échangées donc $a \leq b$ et $a \leq c$ restent vraies.
La troisième propriété s'évalue à `vrai` car la fin coïncide avec la pause 3. On en déduit qu'à la fin $a \leq b \leq c$.

6. Méthode de Hörner

Il est bien évident que i est un variant pour la boucle.
Pour montrer que Φ est un invariant, introduisons des notations. Après l'exécution de $k \in \mathbb{N}$ fois le corps de la boucle, notons i_p l'entier désigné par i, y_p l'objet désigné par y et Φ_p l'évaluation de Φ.
Dans cette convention, y_0 et i_0 représentent les valeurs désignées *avant* la première exécution. Soit $y_0 = a_n$ et $i_0 = n$ ce qui montre que Φ s'évalue à `vrai` à cette étape.
Montrons maintenant que $\Phi_p = \texttt{vrai}$ entraîne $\Phi_{p+1} = \texttt{vrai}$.

$$\left.\begin{aligned} i_{p+1} &= i_p - 1 \\ y_{p+1} &= a_{i_{p+1}} + x * y_p \\ \Phi_p = \texttt{vrai} \Leftrightarrow y_p &= a_{i_p} + \cdots + a_n x^{n-i_p} \end{aligned}\right\} \Rightarrow y_{p+1} = a_{i_{p+1}} + x * y_p$$

d'où $y = a_{i_{p+1}} + a_{i_p} x + \cdots a_n x^{n-i_p+1}$ avec $n - i_p + 1 = n - i_{p+1}$.

7. Tri bulle

L'expression $n-i$ est bien à valeurs dans $\mathbb{N}$ mais elle ne décroît pas strictement. Elle est incrémentée dans le cas de l'échange. Le nombre de montées n'est pas non plus un variant car il est conservé lorsqu'il n'y a pas d'échange. Comment évolue-t-il lors d'un échange ?
Supposons que i désigne i_0 avant un échange. Alors (i_0, i_0+1) est une montée. classons les autres couples susceptibles d'être des montées :

— les couples (k, i_0) et (k, i_0+1) avec $k < i_0$.
— les couples (i_0, l) et (i_0+1, l) avec $i_0 + 1 < l$.
— les couples (k, l) avec $k < i_0$ et $i_0 + 1 < l$.

Pour chaque paire indiquée, le nombre de montées parmi ces couples peut être 0, 1 ou 2 avant l'échange. Il reste identique après l'échange des valeurs. On en déduit que le nombre de montées diminue de 1 par un échange.
Notons m le nombre de montées et considérons $V = n - i + 2m$. Il est bien à valeurs entières. S'il n'y a pas d'échange, i augmente de 1 et m reste identique donc V décroît de 1. S'il y a échange, i augmente de 1 et $2m$ diminue de 2 donc V décroît encore de 1. V est donc bien un variant qui prouve que la boucle se termine.
Considérons la propriété

$$\Phi : \forall k \in \mathbb{N}, 0 \leq k < i \Rightarrow A[k] \geq A[k+1]$$

Elle est vraie après l'initialisation quand i désigne 0 car il n'existe pas de k vérifiant $0 \leq k < i$.
Notons i_0 ce que désigne i avant un échange et supposons que Φ est vraie à ce moment. Après l'échange, i désigne $i_0 - 1$. Comme $k < i_0 - 1$ entraîne $k + 1 < i_0$, les valeurs du tableau intervenant dans Φ ne sont pas modifiés donc Φ reste vraie. Si on incrémente i sans échanger, les deux dernières valeurs de A sont dans le bon ordre donc Φ reste vraie. La propriété Φ est donc un invariant.
À la sortie de la boucle, i désigne n car la condition de réalisation est fausse. L'invariant montre alors que, pour tous les i entre 0 et $n-1$, $A[i+1] \leq A[i]$ c'est à dire que la liste est décroissante.

```
def TriBulle(A):
    n = len(A)
    i = 0
    nbPerm = 0
    while i < n-1:
        if A[i]<A[i+1]:
            A[i],A[i+1] = A[i+1],A[i]
            nbPerm += 1
            if i > 0:
                i -= 1
        else:
            i += 1
    return nbPerm
```

```
A = [1,2,4,5,1,47,52,35,14,41]
print(TriBulle(A))
print(A)
```

8. Développement de Engel

1. La définition de la division euclidienne conduit à l'encadrement $q \leq \frac{b}{a} < q+1$. On en déduit $q = \lfloor \frac{b}{a} \rfloor$.
   La définition de la pseudo division euclidienne conduit à l'encadrement $q-1 < \frac{b}{a} \leq q$. On en déduit $q = \lceil \frac{b}{a} \rceil$.
2. Par définition, le reste d'une pde est strictement plus petit que le nombre par lequel on divise. L'expression a est donc bien un variant.
   Par convention, représentons avec un indice k les valeurs désignées par les noms après l'exécution de k fois le corps de boucle.
   Après l'initialisation et avant la boucle : $s_0 = 0$, $a_0 = a_{ini}$, $Q_0 = 1$ donc $s_0 b + \frac{a_0}{Q_0} = a_{ini}$ ce qui assure que $\Phi_0 =$ `vrai`.
   Après la $k+1$-ème exécution :

$$\left.\begin{aligned} b = q_{k+1}a_k - a_{k+1} \\ Q_{k+1} = Q_k q_{k+1} \\ s_{k+1} = s_k + \frac{1}{Q_k q_{k+1}} \end{aligned}\right\} \Rightarrow s_{k+1}b + \frac{a_{k+1}}{Q_{k+1}} = s_k b + \frac{1}{Q_k q_{k+1}}(b + a_{k+1})$$

$$= s_k b + \frac{1}{Q_k q_{k+1}}(q_{k+1}a_k) = s_k b + \frac{a_k}{Q_k} = a_{ini}$$

   Après la sortie de la boucle a désigne forcément 0 car a désigne toujours un reste de division. Cet algorithme décompose un nombre rationnel en une somme d'inverses.

$$\frac{a_{ini}}{b} = \frac{1}{q_1} + \frac{1}{q_1 q_2} + \cdots$$