Exercice 1. Fonctions avec boucles `for`

Question 1 – Il faut utiliser un `range(1, n+1)`.

Question 2 – Le nombre exact de tours de boucle est `n`. Le nombre de tours de boucle est fini et toutes les opérations sont élémentaires donc la fonction termine pour toute entrée $\texttt{n} \in \mathbb{N}$.

Question 3 – Notre but est de montrer que la fonction renvoie $\sum_{\texttt{k=1}}^{\texttt{n}} \texttt{k}$. Les tours de boucle sont numérotés de `i = 1` à `i = n`. Pour $\texttt{i} \in [\![\texttt{1}, \texttt{n}]\!]$, on note $\texttt{s}_\texttt{i}$ la valeur de la variable `s` à la fin du tour de boucle et :

$$(\mathcal{P}_\texttt{i}) : \texttt{s}_\texttt{i} = \sum_{\texttt{k=1}}^{\texttt{i}} \texttt{k}.$$

Montrons $(\mathcal{P}_\texttt{i})$ par récurrence sur $\texttt{i} \in [\![\texttt{1}, \texttt{n}]\!]$.

Initialisation. À la fin du premier tour de boucle, on a $\texttt{s}_1 = 1$, `i = 1` et donc $\sum_{\texttt{k=1}}^{\texttt{i}} \texttt{k} = 1$. Ainsi, $(\mathcal{P}_1)$ est vrai.

Hérédité. Soit $\texttt{i} \in [\![\texttt{2}, \texttt{n}]\!]$. On suppose $(\mathcal{P}_{\texttt{i-1}})$ on montre $(\mathcal{P}_\texttt{i})$. D'après la ligne 7, on a :

$$\texttt{s}_\texttt{i} = \texttt{s}_{\texttt{i-1}} + \texttt{i} = \texttt{i} + \sum_{\texttt{k=1}}^{\texttt{i-1}} \texttt{k} = \sum_{\texttt{k=1}}^{\texttt{i}} \texttt{k}$$

Donc $(\mathcal{P}_\texttt{i})$ est vraie.

Conclusion. À la fin du dernier tour de boucle, la variable `s` vaut :

$$\texttt{s}_\texttt{n} = \sum_{\texttt{k=1}}^{\texttt{n}} \texttt{k}$$

On en déduit la correction de la fonction :

→ Si `n = 0`, alors la fonction renvoie $\texttt{0} = \sum_{\texttt{k=1}}^{\texttt{0}} \texttt{k}$ (somme vide).

→ Si $\texttt{n} \geqslant \texttt{1}$, alors à la fin du dernier tour de boucle on a `i = n`, donc $\texttt{s} = \sum_{\texttt{k=1}}^{\texttt{n}} \texttt{k}$. Ainsi, la fonction renvoie bien la somme des entiers.

Question 4 – On obtient :

`fibo2` : `range(n-1)` `est_1er` : `range(2,n)` `cpt_1er` : `range(2,n+1)`

Question 5 –

→ `fibo2` : il y a exactement `(n-1)` tours de boucle (et `0` pour `n = 0`).

→ `somme_liste` : il y a exactement `len(L)` tours de boucle.

→ `x_in_L` : il y a au plus `len(L)` tours de boucle.

→ `est_1er` : au plus `(n-2)` tours de boucle.

→ `cpt_1er` : il y a exactement `(n-1)` tours de boucle (et `0` pour `n = 0`).

Question 6 – Tous les invariants sont valables à la fin du tour de boucle.

⋆ `fibo2` : on note $(\mathcal{F}_n)_{n\in\mathbb{N}}$ la suite de Fibonacci :

$$\mathcal{F}_0 = 0 \qquad \mathcal{F}_1 = 1 \qquad \mathcal{F}_{n+2} = \mathcal{F}_{n+1} + \mathcal{F}_n$$

Les tours de boucles sont numérotés de `i = 0` à `i = n - 2`. Pour chaque `i` $\in [\![$`0; n - 2`$]\!]$, on note $\texttt{L}_\texttt{i}$ la valeur de la variable `L` à la fin du tour de boucle numéro `i`. L'invariant de boucle est :

"$\texttt{L}_\texttt{i}$ est de taille `i+3` et pour tout `k` $\in [\![$`0; i + 2`$]\!]$: $\texttt{L}_\texttt{i}\texttt{[k]} = \mathcal{F}_\texttt{k}$".

On en déduit la correction de la fonction :

→ Si `n = 0`, la fonction renvoie `0` $= \mathcal{F}_0$.

→ Si `n = 1`, la fonction renvoie `1` $= \mathcal{F}_1$.

→ Si `n` $\geqslant$ `2`, alors à la fin du dernier tour de boucle, on a `i = n-2`. Ainsi, `L` est de taille `n+1` et pour tout `k` $\in [\![$`0; n`$]\!]$: $\texttt{L}_\texttt{i}\texttt{[k]} = \mathcal{F}_\texttt{k}$. Finalement, la fonction renvoie `L[n]` qui existe et vaut bien $\mathcal{F}_\texttt{n}$.

⋆ `x_in_L` : on numérote les tours de boucle par `i = 0, 1, 2...`

"Si le tour numéro `i` est exécuté sans retour anticipé, alors `x` $\notin \left\{\texttt{L[k] : k} \in [\![\texttt{0; i}]\!]\right\}$".

où `i` $\in [\![$`0, len(L) - 1`$]\!]$ est le numéro du tour de boucle. On en déduit la correction de la fonction :

→ Si la fonction renvoie `True`, c'est que la ligne 5 a été exécutée. Par la ligne 4 : `x == L[i]` et donc `x` appartient à `L`.

→ Si la fonction renvoie `False`, alors :

- Si `n = 0`, alors `L = []` et la fonction renvoie `False` ce qui est correct (aucun `x` n'appartient à `[]`).
- Si `n` $\geqslant$ `1`, alors à la fin du dernier tour de boucle : `i = n - 1`. Donc d'après l'invariant de boucle, `x` n'appartient pas à $\left\{\texttt{L[k] : k} \in [\![\texttt{0; n - 1}]\!]\right\}$. On en déduit que `x` n'appartient pas à `L`.

Finalement, la fonction renvoie `True` si `x` appartient à `L` et `False` sinon.

⋆ `somme_liste` : on numérote les tours de boucle de `i = 0` à `i = n-1` où `n = len(L)`. Pour chaque `i` $\in [\![$`0; n - 1`$]\!]$, on note $\texttt{s}_\texttt{i}$ la valeur de la variable `s` à la fin du tour de boucle numéro `i`. L'invariant de boucle est :

$$\texttt{s}_\texttt{i} = \sum_{\texttt{k=0}}^{\texttt{i}} \texttt{L[k]}.$$

Pour la correction, on utilise le même raisonnement que pour la fonction `somme`.

⋆ `est_1er`. Les tours de boucle sont numérotés par `d = 2, 3, 4...`. L'invariant de boucle est :

"Si le tour numéro `d` est exécuté sans retour anticipé, alors aucun entier dans $[\![$`2, d`$]\!]$ ne divise `n`".

où `s` $\in [\![$`2, n - 1`$]\!]$ est le numéro du tour de boucle.
Pour la correction, on utilise le même raisonnement que pour la fonction `x_in_L`.

⋆ `cpt_1er`. Les tours de boucle sont numérotés de `i = 2` à `i = n`. Pour chaque `i` $\in [\![$`2; n`$]\!]$, on note $\texttt{cpt}_\texttt{i}$ la valeur de la variable `cpt` à la fin du tour de boucle numéro `i`. L'invariant de boucle est :

"$\texttt{cpt}_\texttt{i}$ est le nombre de nombres premiers dans l'intervalle $[\![$`2, i`$]\!]$".

Pour la correction, on utilise le même raisonnement que pour la fonction `somme`.

Exercice 2. Fonctions avec boucles `while`

Question 1 – La quantité `v = len(L) - k - 1` est un variant de boucle, en effet `v` $\in \mathbb{N}$ au début de chaque boucle (par la condition de la boucle `while`) et `v` décroît strictement à chaque tour. Puisqu'il existe un variant, le nombre de tours de boucle est fini. Toutes les opérations sont élémentaires, donc la fonction `ind` termine.

Question 2 – Montrons que la fonction renvoie le premier indice `k` tel que `L[k] = x` et `None` si `x` n'appartient pas à `L`. Soit `N` le nombre de tours de boucle et numérotons les tours par `i = 0, 1, 2..., N - 1`. On remarque que la variable `k` vaut `i` au début du tour de boucle numéro `i` et `i+1` à la fin. Pour chaque `i` $\in [\![0 ; N-1]\!]$, posons :

$$(\mathcal{P}_i) \text{ : si le tour de boucle } i \text{ existe, alors } x \text{ n'appartient pas à } \left\{ L[k] \text{ : } k \in [\![0 ; i]\!] \right\}.$$

Initialisation. Au début du premier tour de boucle : `i = 0`. Donc `L[0:i+1]` est la liste qui ne contient que `L[0]`. Or `x` n'est pas égal à `L[0]` d'après la condition de la boucle `while`. Donc $(\mathcal{P}_0)$ est vraie au premier tour de boucle.

Hérédité. Soit `i` $\in [\![1, N-1]\!]$. On suppose $(\mathcal{P}_{i-1})$ et on montre $(\mathcal{P}_i)$. Par $(\mathcal{P}_{i-1})$, `x` n'appartient pas à `L[0:i]`. De plus, d'après la condition de la boucle `while`, `L[i] != x`. On en déduit $(\mathcal{P}_i)$.

Conclusion. Au début du dernier tour de boucle, la valeur de la variable `k` était `N - 1`. Ainsi, d'après $(\mathcal{P}_{N-1})$, `x` n'appartient pas à `L[0 : N]`. Notez que si `N = 0` alors $(\mathcal{P}_{N-1})$ n'est pas définie, mais on a quand même la propriété "`x` n'appartient pas à `L[0 : N] = []`".
D'après la condition de la boucle `while`, on sort de la boucle `while` pour l'une des deux raisons suivantes :

- → `N = len(L)`. Dans ce cas, l'invariant de boucle indique que `x` n'appartient pas à `L`. La fonction renvoie `None` qui est le résultat attendu.
- → `L[N] = x`. D'après $(\mathcal{P}_{N-1})$, `N` est le premier indice tel que `L[N] = x`. La fonction renvoie `N` qui est le résultat attendu.

En conclusion la fonction est correcte.

Question 3 – La ligne 5 est `while i < len(L)`.

Question 4 – La quantité `v = len(L) - i` est un variant de boucle.

Question 5 – On numérote les tours de boucle de `j = 0` à `j = len(L)-1`. Notons que la variable `i` vaut `j` au début du tour de boucle numéro `j` et `j+1` à la fin.

Pour chaque `j` $\in [\![0, \texttt{len(L)} - 1]\!]$, on note s_j la valeur de la variable `s` à la fin du tour de boucle d'indice `j`. La proposition suivante est un invariant de boucle :

$$s_j = \sum_{k=0}^{j} L[k]$$

Exercice 3. Fonctions récursives

Question 1 –

```
1 # On suppose que n >= 0
2 def nb_chiffres(n):
3     """(n: int) -> int"""
4     if n == 0:
5         return 0
6     else:
7         return 1 + nb_chiffres(n//10)
```

Question 2 –

Terminaison. On le montre par récurrence sur `n` $\in \mathbb{N}$.

Initialisation. Si `n = 0` alors l'appel à `nb_chiffres(n)` termine d'après les lignes 4 et 5.

Hérédité. Si `n > 0`, on suppose que l'appel à `nb_chiffres(m)` termine pour tout `m < n`. L'appel à `nb_chiffres(n)` fait un appel à `nb_chiffres(n//10)` qui termine par hypothèse de récurrence. Toutes les autres opérations sont élémentaires, donc l'appel termine. En conclusion, l'appel à `nb_chiffres(n)` termine pour tout `n` $\in \mathbb{N}$.

Correction. On le montre par récurrence sur `n` $\in \mathbb{N}$.

Initialisation. Si `n = 0` alors l'appel à `nb_chiffres(n)` renvoie 0 d'après les lignes 4 et 5 ce qui est correct.

Hérédité. Si `n > 0`, on suppose que l'appel à `nb_chiffres(m)` est correct pour tout `m < n`. L'appel à `nb_chiffres(n)` renvoie le nombre de chiffres dans `n//10` par hypothèse de récurrence. On note `c` ce nombre de chiffres, alors l'appel principal renvoie `c + 1` qui est bien le nombre de chiffres dans `n`. En conclusion, l'appel à `nb_chiffres(n)` renvoie le nombre de chiffres dans `n` pour tout `n` $\in \mathbb{N}$.

Exercice 4. Miroir d'une liste

Question 1 – On fait des exemples pour trouver la formule générale :

- → `len(L) = 2` ⇝ `range(0, 1)`
- → `len(L) = 3` ⇝ `range(0, 1)`
- → `len(L) = 4` ⇝ `range(0, 2)`
- → `len(L) = 5` ⇝ `range(0, 2)`
- → `len(L) = 6` ⇝ `range(0, 3)`

Formule générale : `range(0, len(L)//2)`.

Question 2 – Le nombre de tours de boucle est :

$$\left\lfloor \frac{\texttt{len(L)}}{2} \right\rfloor.$$

La fonction `swap` est composée d'opérations élémentaires donc elle termine sous l'hypothèse que `i` et `j` sont des indices valides pour `L`. Le nombre de tours de boucle dans la fonction `miroir` est fini, cette fonction fait appel à `swap` qui termine et les autres opérations sont des opérations élémentaires, donc la fonction `miroir` termine pour toute entrée `L`.

Question 3 – On remarque que la fonction `swap` inverse les éléments d'indices `i` et `j` dans `L`. Pour `i` $\in$ `{0, ..., len(L)//2 - 1}`, on note $\texttt{L}_\texttt{i}$ la valeur de la variable `L` à la fin du tour de boucle. On note $(\mathcal{P}_\texttt{i})$ la proposition suivante :

$$(\mathcal{P}_\texttt{i}) : \begin{cases} \text{La liste } \texttt{L}_\texttt{i} \text{ est égale à } \texttt{L}_\texttt{0} \text{ où pour tout } \texttt{k} \in [\![\texttt{0, i}]\!] \text{ les éléments d'indices } \texttt{k} \text{ et} \\ \texttt{(len(L) - k - 1)} \text{ ont été échangés.} \end{cases}$$

La proposition $(\mathcal{P}_\texttt{i})$ se montre par itération finie sur i. On en conclut que la fonction `miroir` est correcte.

Exercice 5. Tri par sélection

Question 1 – La commande `print(tri([5,6,7,0,2]))` affiche ce que renvoie la fonction, c'est à dire `None`. Ça ne donne aucune indication sur la façon dont a été modifiée la liste. Pour tester la fonction, on peut exécuter le programme suivant :

```
1 print(L)
2 tri(L)
3 print(L)
```

Question 2 – La fonction `mini` renvoie l'indice d'un élément minimum de `L[i0:]`. La fonction marche lorsque `i0` est un indice valide pour `L`.

Terminaison. Cette fonction termine car la quantité `v = len(L) - i` est un variant de boucle.

Correction. Pour $\texttt{i} \in \{\texttt{i0 + 1}, \dots, \texttt{len(L) - 1}\}$, on note $\texttt{m}_\texttt{i}$ la valeur de la variable `m` au début du tour de boucle et $(\mathcal{P}_\texttt{i})$ la proposition : $\texttt{L[m}_\texttt{i}\texttt{]}$ est le minimum de `L[i0:i]`. Montrons que $(\mathcal{P}_\texttt{i})$ est un invariant de boucle par récurrence sur $\texttt{i} \in \{\texttt{i0 + 1}, \dots, \texttt{len(L) - 1}\}$:

Initialisation. Au premier tour de boucle : `i = i0 + 1` et $\texttt{m}_\texttt{i} = \texttt{i0}$ donc $\texttt{L[m}_\texttt{i}\texttt{]}$ est bien le minimum de `L[i0:i]` dont le seul élément est `L[i0]`.

Hérédité. Supposons $(\mathcal{P}_\texttt{i})$ et montrons $(\mathcal{P}_{\texttt{i+1}})$. D'après les lignes 6 et 7, on a deux cas :

- → Si $\texttt{L[m}_\texttt{i}\texttt{]} > \texttt{L[i]}$ alors $\texttt{m}_{\texttt{i+1}} = \texttt{i}$. Or d'après $(\mathcal{P}_\texttt{i})$, $\texttt{L[m}_\texttt{i}\texttt{]}$ est le minimum de `L[i0 : i]`. On en conclut que $\texttt{L[m}_{\texttt{i+1}}\texttt{]}$ est inférieur strictement à tous les éléments de `L[i0 : i]`. Donc $(\mathcal{P}_{\texttt{i+1}})$ est vraie.
- → Sinon $\texttt{L[m}_\texttt{i}\texttt{]} \leq \texttt{L[i]}$ et $\texttt{m}_{\texttt{i+1}} = \texttt{m}_\texttt{i}$. Or d'après $(\mathcal{P}_\texttt{i})$, $\texttt{L[m}_\texttt{i}\texttt{]}$ est le minimum de `L[i0 : i]`. On en conclut que $\texttt{L[m}_{\texttt{i+1}}\texttt{]}$ est inférieur ou égal à tous les éléments de `L[i0 : i + 1]`. Donc $(\mathcal{P}_{\texttt{i+1}})$ est vraie.

Question 3 –

Terminaison. On remarque que la boucle `for` effectue `len(L)` tours et que les appels aux fonctions `mini` et `swap` terminent. Donc la fonction `tri` termine.

Correction. On remarque que la fonction `swap` inverse les éléments d'indices `i` et `j` dans `L`.
Pour $\texttt{i}_0 \in \{\texttt{0}, \dots, \texttt{len(L)-1}\}$, on note $\texttt{L}_{\texttt{i}_0}$ la valeur de la variable `L` au début du tour de boucle d'indice $\texttt{i}_0$ et $(\mathcal{P}_{\texttt{i}_0})$ la proposition suivante :

$$\begin{cases} \texttt{L}_{\texttt{i}_0} \text{ et } \texttt{L}_0 \text{ contiennent les mêmes éléments} \\ \texttt{L}_{\texttt{i}_0}\texttt{[0 : i}_0\texttt{]} \text{ contient les } \texttt{i}_0 \text{ plus petits éléments de } \texttt{L}_0 \text{ triés par ordre croissant} \end{cases}$$

Montrons que $(\mathcal{P}_{\texttt{i}_0})$ est un invariant de boucle par récurrence sur $\texttt{i}_0 \in \{\texttt{0}, \dots, \texttt{len(L)-1}\}$:

Initialisation. Montrons $(\mathcal{P}_{\texttt{i}_0})$ pour $\texttt{i}_0 = \texttt{0}$. $\texttt{L}_{\texttt{i}_0}\texttt{[0 : i}_0\texttt{]}$ est la liste vide donc $(\mathcal{P}_0)$ est vraie.

Hérédité. On suppose $(\mathcal{P}_{\texttt{i}_0})$ et on montre $(\mathcal{P}_{\texttt{i}_0\texttt{+1}})$. D'après les lignes 4 et 5 :

$$\begin{cases} \texttt{L}_{\texttt{i}_0\texttt{+1}} \text{ et } \texttt{L}_{\texttt{i}_0} \text{ contiennent les mêmes éléments} \\ \texttt{L}_{\texttt{i}_0\texttt{+1}}\texttt{[i}_0\texttt{]} \text{ est l'élément minimum de } \texttt{L}_{\texttt{i}_0}\texttt{[i}_0\texttt{ :]} \end{cases}$$

En combinant avec $(\mathcal{P}_{\texttt{i}_0})$, on en déduit $(\mathcal{P}_{\texttt{i}_0\texttt{+1}})$.

Conclusion. À la sortie de la boucle `for`, la proposition $\mathcal{P}_{\texttt{len(L)}}$ est vraie, c'est à dire que `L[0 : len(L)]` contient les `len(L)` plus petits éléments de L_0 triés par ordre croissant. Finalement, la fonction `tri` est correcte.