1 Parcours séquentiel d'un tableau :

Problème : Je recherche un fichier nommé "algo.pdf" que je pense avoir rangé dans un dossier nommé INFO.

Pour simplifier nous supposons que ce dossier est un tableau T de chaînes de caractères ; pour chercher ce fichier on va utiliser l'algorithme suivant :

Algorithme 1 : Recherche séquentielle

Données : Un tableau T ayant n éléments de même type et comparables et une valeur v
Résultat : Un entier i, la première position de la valeur dans le tableau

```
1 début
2 |  i ← 0
3 |  tant que  T[i] <> v faire
4 |  |  i ← i + 1
5 |  fin
6 fin
```

1. Les **spécifications** d'un algorithme est l'ensemble des **Données** et du **Résultat** d'un algorithme. En quelque sorte c'est une sorte de **contrat** associé à l'algorithme .

   Attention rien n'assure que ce contrat sera respecté lors de l'exécution de l'algorithme.

   Est ce que les spécifications sont suffisamment précises ?
2. Appliquer l'algorithme pour T = ["olga.txt", "algo.py","goal.csv",algo.pdf", "galo.py"] et pour la valeur v = "algo.pdf"

   Quel est la valeur de i à la fin de l'algorithme ?
3. Que se passe-t-il cette fois-ci si T = ["olga.txt", "algo.py","goal.csv", "galo.py"] ? Corriger l'algorithme
4. Implémenter cet algorithme en Python sous la forme d'une fonction qui renvoie l'entier i
5. Faire une deuxième implémentation en Python avec une boucle for et un retour dans la boucle.
6. Que se passe-t-il si le tableau est trié en ordre croissant (ordre lexicographique) ?

2 Notion de complexité :

Algorithme 2 : Recherche séquentielle

Données : Un tableau T ayant n éléments et une valeur v
Résultat : Un entier i, la première position de la valeur dans le tableau

```
1  début
2      i ← 0
3      tant que (i < n) et (T[i] <> v) faire
4          i ← i + 1
5      fin
6      si i = n alors
7          afficher("v n'est pas dans le tableau T")
8      sinon
9          afficher("v est dans le tableau en position",i)
10     fin
11 fin
```

Lorsqu'on exécute ce programme pour une liste de 10 éléments on a l'impression qu'il s'exécute presqu'instantanément.

Que se passe-t-il si la liste contient 100 000 ou 1 million d'éléments ?

On aimerait pouvoir répondre à cette question en ayant un ordre de grandeur du lien entre le temps d'exécution de l'algorithme T et la taille n du tableau

Dans la boucle `Tant que` deux opérations sont répétées : une *affectation* et deux *comparaisons*.

On peut supposer que le temps d'exécution d'une affectation peut varier d'un tour de boucle à l'autre mais ne dépassera pas une constante c_1, de même pour la comparaison le temps d'exécution d'une comparaison est plus petite qu'une constante c_2 à chaque tour de boucle. Combien de fois sera exécuté la boucle ?

On envisage deux cas extrêmes :

- **Au pire des cas** l'élément cherché est à la fin du tableau ou ne se trouve pas dans le tableau et dans ce cas le temps d'exécution $T(n) \leqslant nc_1 + 2nc_2 = (c_1 + 2c_2)n$. On dit que la complexité est linéaire dans le pire des cas, on dit aussi que la complexité est en O(n) (lire en grand O de n)
- **Au meilleur des cas** l'élément cherché est au début du tableau dans ce cas le temps ne dépend pas de n on dit que la complexité est en O(1)

3 Exercices - Parcours séquentiel d'un tableau - Complexité

Ex 1

Si sur une machine particulière il a fallu 1 s pour trouver un élément d'un tableau de taille 100 000 en dernière position, combien de temps à peu près mettra -t-on pour trouver un élément en dernière position dans un tableau de taille 1 000 000 sur cette même machine ?

Ex 2

Modifier l'algorithme de recherche séquentielle afin d'avoir comme information tous les indices i tel que $T[i] = v$. Quelle est la complexité de cet algorithme dans tous les cas ?

Ex 3

Compléter la fonction `somme (liste)` qui prend en argument une liste non vide d'entiers et renvoie la somme de ses éléments.

Compléter aussi le programme principal

Algorithme 3 : Somme des entiers contenus dans une liste

```
début
    /* Définition de la fonction somme(liste)                    */
1   somme (T)
2   début
        Données : Un tableau T non vide ayant n entiers
        Résultat : la somme des n entiers
3       somme ← 0
4       pour  i ← 0 jusqu'à n − 1 faire
5           .....
6           i ← i + 1
7       fin
8       retourner .....
9   fin
    /* ----Programme Principal ------------                      */
10  liste ← [1,6,6,4]
11  .....
fin
```

1. Quel est le nombre d'additions ? En déduire la complexité
2. Ecrire une fonction `moyenne (liste)` qui prend en argument une liste non vide d'entiers et renvoie la moyenne de ses éléments. On pourra utiliser la fonction `somme(liste)` ci-dessus.

Ex 4

1. On définit en mathématiques la moyenne harmonique de n nombres x_1, x_2,...,x_n comme étant l'inverse de la moyenne des inverses de ces n nombres. Ecrire une fonction `moyenneHarmonique(liste)` qui prend en argument une liste non vide d'entiers et renvoie la moyenne harmoniquede ses éléments. On pourra utiliser la fonction `moyenne(liste)` ci-dessus.

2. On définit en mathématiques la moyenne pondérée de n nombres x_1, x_2,...,x_n par des coefficients c_1, c_2, ..., c_n par :
$$\frac{c_1 \times x_1 + c_2 \times x_2 + ... + c_n \times x_n}{c_1 + c_2 + ... + c_n}$$
   (a) Ecrire une fonction `moyennePonderee(valeurs,coefficients)` qui prend en argument, une liste non vide de nombres `valeurs` et une liste non vide d'entiers `coefficients` et qui renvoie la moyenne pondérée .
   (b) Implémenter cette fonction en Python
   (c) Calculer la complexité en temps de cette fonction en prenant uniquement en compte les multiplications

Exercices - Recherche d'un Maximum

Ex 5

On a une liste L de nombres décimaux, les températures maximales relevées à la station météo de Vélizy Villacoublay pour chaque jour du mois de Juin 2019

1. Ecrire une fonction en pseudo-code retournant le jour de la température maximale sur le mois de Juin 2019. Bien préciser les spécifications de la fonction
2. Ecrire une deuxième fonction retournant à la fois le jour et la valeur de la température maximale sur le mois de Juin 2019

Ex 6

Cette fois-ci chaque élément de la liste est une liste à deux éléments contenant la température minimale de la journée en premier, puis la température maximale de la journée ". Modifier la fonction de telle sorte que l'on ait le jour du mois de Juin où la température minimale a été la plus basse ainsi que la valeur de ce minimum, et de même pour la température maximale

Ex 7

Quelle est la complexité dans le pire des cas de la recherche d'un maximum dans une liste ?

Ex 8

Si on sait qu'une liste est **déjà** triée dans l'ordre **croissant** que suffit-il de faire pour avoir le maximum de la liste ? Quelle est alors la complexité ?

Ex 9

Une liste contient des entiers non distincts mais triés dans l'ordre croissant par exemple [1,1,2,2,2,3]
De cette liste on veut créer une nouvelle liste ne contenant qu'un exemplaire de la précédente liste mais avec un seul exemplaire des entiers ici [1,2,3]
Donner un algorithme et évaluer sa complexité