Exercice n°1

Il s'agit de lancer 4 threads et d'attendre la fin (après le lancement des 4 threads qui devront tourner **en parallèle**) afin d'afficher le retour.

Le thread lancé comptera le nombre de fois qu'un mot cible (ex : « printf ») apparaît dans un fichier.

Dans un tout premier temps (**première version**), le nom du fichier (et le nombre de tabulations, voir plus bas) sera défini en local dans le thread. Dans ce cas, il y a 4 threads « différents » et le code de retour correspond au nombre d'occurrences du mot cible dans le fichier.

Le programme affichera alors, pour chaque thread, le résultat demandé.

Il est bien évident que très rapidement (**seconde version**), le nom du fichier et le nombre de tabulations seront transmis par paramètre au thread, et que le thread sera lancé 4 fois.

Dans le seul but de prendre du temps d'exécution,
Le thread, dans une boucle (i étant initialisé à 0),

- ouvre le fichier,
- se positionne sur le caractère de position i dans le fichier,
- lit autant de caractères que nécessaire (ex : 6 si mot cible = « printf ») (attention à la fin de fichier !),
- ferme le fichier,
- teste si la chaîne lue correspond au mot cible et incrémente éventuellement un compteur.
- Incrémente i de **un**, remonte dans la boucle et cela jusqu'à la fin du fichier.

Toujours pour prendre du temps, et mieux se rendre compte de l'ordre d'exécution des threads, à chaque passage dans la boucle et plus précisément à chaque ouverture de fichier, chaque thread affichera une « * » précédée de 0, 1, 2 ou 3 tabulations selon le thread.

On obtiendra donc un résultat du type suivant

```
*
            *
*
                        *
                                    *
            *
*
                        *
                                    *
                        *
             *
*


…
et ainsi de suite
```