

# R1.01 – TD7

## Objectif : comptage des mots dans une chaîne de caractères

Dans cet exercice un mot est défini comme une suite **non vide** de lettres (minuscules ou majuscules), de chiffres et ou de caractère "souligné" '\_' (**underscore**).

Pour cela vous disposer de la fonction de signature suivante :

```
fonction compterMot (chaîne : in string) renvoie
entier_naturel ;
```

## Exercice 1

Ecrire l'algorithme qui saisit au clavier une suite de lignes (des `strings`), jusqu'à ce que l'utilisateur tape une ligne vide (appuie sur la touche `Entrée` sans avoir saisi aucun caractère).

Au fur et à mesure de la saisie :

- afficher la chaîne lue sur une nouvelle ligne,
- afficher sur la ligne suivante le nombre de mots de la ligne lue (en faisant appel à la fonction `compterMot()`), précédé de la chaîne
- Nombre de mots de la ligne :
- sauter une ligne.

## Description des approches

Il ne s'agit pas d'identifier complètement les mots mais seulement de les compter.

Or il y a autant de mots que de débuts de mots ! Il suffit donc de compter les débuts de mots.

Le début d'un mot peut être défini comme une lettre (majuscule ou minuscule), un chiffre ou un '\_', précédé par un caractère qui n'est aucun de ces caractères.

Un mot peut commencer au rang 0. Il ne faut donc pas commencer la boucle de traitement au caractère de rang 1. On ne peut pas non plus comparer le premier caractère (de rang 0) avec son prédécesseur.

Il y a donc deux possibilités : soit traiter à part le premier caractère, ce qui est lourd, soit comparer le caractère courant avec son prédécesseur, qui aura été mémorisé dans une variable intermédiaire et qui, pour le rang 0, aura correctement été initialisé.

Dans les exercices suivants, nous allons coder la fonction `compterMot ()` selon les deux approches proposées

## Exercice 2 Approche caractère/caractère

L'algorithme de haut niveau, dans ce cas, est le suivant :

```
....
se_positionner_en_debut_de_chaine;
pour (chaque_caractere_de_la_chaine)
    faire
        si (c_est_un_debut_de_mot)
            compter_le_mot;
        fsi
    passer_au_caractere_suivant;
ffaire
....
```

Dans cet algorithme, la chaîne est vue comme une simple suite de caractères, qui sont donc traités séquentiellement.

## Exercice 3 Approche Approche mot/mot

L'algorithme de haut niveau, dans ce cas, est le suivant :

```
....
se_positionner_en_debut_de_mot;
jusqua (fin_de_ligne)
    faire
        compter_le_mot;
        passer_a_la_fin_du_mot;
        se_positionner_en_debut_de_mot;
    ffaire
....
```

Dans ce cas, l'algorithme superpose à la structure **physique** du tableau vu comme une "suite de caractères", une structure **logique** de "suite de mots".

## Exercice 4

A la définition précédente d'un mot ("*un mot est défini comme une suite **non vide** de lettres (minuscules ou majuscules), de chiffres et ou de caractère "souligné" ' \_ '*") est ajoutée une contrainte supplémentaire : *un mot doit **commencer** par une lettre (minuscule ou majuscule).*

Dans la ligne suivante :

A440 123 \_B732 44 \_25

il n'y a que **deux** mots : A440 et B732.

Modifier les deux fonctions codées précédemment en conséquence.