

Exercice 1.

```
#include <stdio.h>
#include <stdlib.h>
#include <stdbool.h>
#include <ctype.h>

static int
nombre_lettres (const char s[])
{
    int nb= 0;
    for(int i= 0; s[i] != '\0'; i++)
        if (isalpha(s[i]))
            nb++;
    return nb;
}

static void
usage(char s[])
{
    fprintf(stderr, "Usage : %s \"chaîne\"\n", s);
    exit(EXIT_FAILURE);
}

int
main(int argc, char *argv[])
{
    if (argc != 2)
        usage(argv[0]);

    printf("%s\t%d\n", argv[1], nombre_lettres(argv[1]));
    return EXIT_SUCCESS;
}
```

Quel est le résultat de l'exécution de ce programme, de nom `tester-chaîne` dans les cas suivants :

```
$ ./tester-chaîne
$ ./tester-chaîne Hello,World!
```

Note : `int isalpha (int c)`; est déclaré dans `ctype.h`
`isalpha()` vérifie si l'on a un caractère alphabétique.

Exercice 2. Complétez le programme précédent avec les fonctions suivantes :

- `int occurrence_caractere(const char *s, char c)` qui calcule et retourne le nombre d'occurrences du caractère `c` dans la chaîne `s`.
Par exemple `occurrence("croco", 'c')` retourne 2.
- `double frequence_caractere(const char *s, char c)` qui calcule la fréquence des occurrences du caractère `c` dans la chaîne `s`.
Rappel : la fréquence est égale au quotient du nombre d'occurrences par le nombre de caractères de la chaîne. Par exemple `frequence("croco", 'c')` renvoie 0.4.
- `int nombre_majuscules(const char *s)` qui calcule et retourne le nombre de lettres en majuscule dans la chaîne `s`.
- `int nombre_séparateurs(const char *s)` qui calcule et retourne le nombre de séparateurs dans la chaîne `s`. Les séparateurs sont les caractères `'\t'` `'\n'` `'\v'` `'\f'` `'\r'` `' '`.
- `void chaîne_en_majuscule(char *s)` qui met la chaîne `s` en majuscules.

- `int chaine_compare(const char *s1, const char *s2)`. Il est demandé de reprogrammer `strcmp` (voir la documentation).
- `int chaine_casse_compare(const char *s1, const char *s2)` similaire à `chaine_compare` mais en ignorant la différence entre majuscules et minuscules.

Exercice Complémentaire 1. Écrire une fonction `int compter_mots(const char *s)` qui découpe la chaîne `s` en mots en utilisant les séparateurs et renvoie le nombre de mots ainsi obtenus. Les séparateurs sont les caractères `'\t'` `'\n'` `'\v'` `'\f'` `'\r'` `' '`.

Par exemple `compter_mots("ceci est un exemple test")` renvoie 5.

Attention deux séparateurs consécutifs ne créent pas un nouveau mot. Il en est de même pour un séparateur situé en début ou en fin de chaîne.