Cours de programmation séquentielle

Chaînes de caractères

1 Buts

- Utilisation des tableaux unidimensionnels.
- Utilisation des chaînes de caractères.
- Écriture de librairie de fonctions.

2 Énoncé

Une chaîne de caractère est un tableau unidimensionnel de `char` se terminant par le caractère `'\0'` ou l'entier `0`. Le but de ce travail pratique est d'écrire une librairie `mystring.h` permettant de manipuler des chaînes de caractères. Puis dans un deuxième temps, il s'agira d'écrire un programme démontrant des cas d'utilisations votre librairie.

2.1 La librairie `mystring.h`

Votre librairie doit contenir les fonctions suivantes:

1. Ajoute la chaîne de caractère `src` à la fin de la chaîne de caractères `dest`. Cette fonction retourne `0` si tout s'est bien passé, `1` si une erreur s'est produite (p.ex. la chaîne `dest` est trop petite). Ici `size` est la capacité de la chaîne `dest`.

   ```
   int string_cat(size_t size, char *dest, char *src);
   ```

2. Retourne un pointeur vers la première occurrence du caractère `c` dans la chaîne de caractère `s`. Si le caractère est absent on retourne `0`.

   ```
   char *string_chr(char *s, char c);
   ```

3. Détermine l'égalité deux chaînes de caractères

   ```
   bool string_cmp(char *s1, char *s2);
   ```

4. Copie la chaîne de caractères `src` dans la chaîne de caractères `dest`. Cette fonction retourne `0` si tout s'est bien passé, `1` si une erreur s'est produite (p.ex. la chaîne `dest` est trop petite). Ici `size` est la capacité de la chaîne `dest`.

   ```
   int string_cpy(size_t size, char *dest, char *src);
   ```

5. Retourne la longueur de la chaîne de caractères `s`. On le compte pas le `\0` terminal dans ce décompte.

   ```
   size_t string_len(char *s);
   ```

6. Ajoute au plus `n` caractères de la chaîne de caractères `src` dans la chaîne de caractères `dest`. Cette fonction retourne `0` si tout s'est bien passé, `1` si une erreur s'est produite (p.ex. la chaîne `dest` est trop petite).

   ```
   int string_ncat(size_t size, char *dest, char *src, size_t n);
   ```

7. Détermine l'égalité d'au plus `n` octets des chaînes de caractères `s1` et `s2`.

   ```
   bool string_ncmp(char *s1, char *s2, size_t n);
   ```

8. Copie au plus `n` caractères de la chaîne de caractères `src` à la chaîne de caractères `dest`. Cette fonction retourne `0` si tout s'est bien passé, `1` si une erreur s'est produite (p.ex. la chaîne `dest` est trop petite).

   ```
   int string_ncpy(char *dest, char *src, size_t n);
   ```

9. Retourne le nombre d'occurrences du caractère `c` dans la chaîne de caractères `src`.

   ```
   size_t string_cnt_chr(char *src, char c);
   ```

10. Échange aléatoirement de place les caractères contenus dans la chaîne de caractères `src`

    ```
    void string_fry(char *src);
    ```

2.2 Remarque

On alloue ici des tableaux statiques, dont la taille est connue à la compilation, p.ex. une taille de `100` pour la taille maximale des tableaux de `char` à utiliser pour stocker les chaînes de caractères (pensez à utiliser le préprocesseur). Idéalement, ces fonctions utilisent l'allocation dynamique de mémoire, mais cela est pour une prochaine fois.

2.3 Programme

Écrire un programme, `string_manip.c`, qui utilise les fonctions de votre librairie et démontre son bon fonctionnement en affichant les résultats à l'écran.

2.4 Compilation

Pour compiler vous devez écrire un `Makefile` contenant au moins deux cibles:

- une cible `string_manip` qui compile votre programme principal et produit l'exécutable `string_manip`.
- une cible `clean` qui efface tous les produits de compilation.