Erick Gallesio, S. Lavirotte
Polytech Nice Sophia
SI3

TD séances n° 3
Chaînes de caractères

Dans les exercices de cette feuille, on construit des fonctions de traitement sur des chaînes de caractères (des tableaux de caractères qui se terminent par le caractère '`\0`').

1 Recherche de caractère

On veut construire deux fonctions permettant de chercher un caractère dans une chaîne de caractères.

1. Construire la fonction

```
int indice(const char str[], const char c);
```

qui renvoie le plus petit indice où se trouve le caractère `c` dans la chaîne `str`. Par convention, cette fonction renvoie `-1` si le caractère `c` n'apparaît pas dans `str`.

2. Construire la fonction

```
int indice_droite(const char str[], const char c);
```

qui renvoie le plus grand indice où se trouve le caractère `c` dans la chaîne `str`. Par convention, cette fonction renvoie `-1` si le caractère `c` n'apparaît pas dans `str`.

On a donc les résultats suivants:

```
indice("Test", 'T')          ⇒ 0
indice("Test", 't')          ⇒ 3
indice("Test", 'z')          ⇒ -1
indice("Tester", 'e')       ⇒ 1

indice_droite("Test", 'T')   ⇒ 0
indice_droite("Test", 't')   ⇒ 3
indice_droite("Test", 'z')   ⇒ -1
indice_droite("Tester", 'e') ⇒ 4
```

2 Palindrome[1]

Écrire la fonction

```
int palindrome(const char str[]);
```

qui renvoie `1` si la chaîne `str` est un palindrome et `0` sinon.

```
palindrome("ressasser") → 1
palindrome("kayak")     → 1
palindrome("X")         → 1
palindrome("test")      → 0
```

3 Fonctions standard sur chaînes

Réécrire les fonctions standard suivantes:

```
void strcpy(char s1[], char s2[]);
int  strcmp(char s1[], char s2[]);
void strupper(char s[]);
```

où

strcpy:

recopie les caractères de la chaîne `s2` à la place de ceux de de la chaîne `s1`.

[1] https://fr.wikipedia.org/wiki/Palindrome

TD séances n° 3
Chaînes de caractères

strcmp:

compare les chaînes de caractères `s1` et `s2`. Cette fonction renvoie:

- `0` si les deux chaînes sont identiques.
- une valeur négative si `s1` < `s2`
- une valeur positive si `s1` > `s2`

Attention: La comparaison se fait ici sur l'ordre lexicographique (et non pas sur la taille des chaînes). On a donc

- "abc" < "z"
- "abc" < "abd"
- "trié" < "triée"

strupper :

convertit la chaîne de caractères en majuscules (utiliser pour cela votre propre fonction de conversion).

Ecrire un programme permettant de tester ces fonctions.

4 Suppression d'un caractère

Écrire la fonction

```
void suppress_char(char str[], char c);
```

qui permet de supprimer toutes les apparitions du caractère `c` dans la chaîne `str`. Pour écrire cette fonction, vous essaierez de **ne pas utiliser de chaîne temporaire**.

5 Une version simplifiée de fgrep

Sous Unix, la commande

```
$ fgrep chaîne < fichier
```

permet d'afficher les lignes de fichier qui contiennent la séquence chaîne.

1. Pour commence, coder la fonction

```
int Strstr(char a[], char b[]);
```

qui renvoie

- `1` si la sous-chaîne `b` apparaît à l'intérieur de la chaîne `a`, ou
- `0` si `b` n'est pas une sous-chaîne de `a`.

2. Utiliser cette fonction pour écrire un programme qui, étant donné une chaîne en argument, affiche les lignes de son entrée qui la contiennent. Pour accéder au premier argument passé sur la ligne de commande, déclarez votre fonction `main` de la façon suivante:

```
int main(int argc, char *argv[])
```

Le premier paramètre de la ligne de commande se trouve dans `argv[1]`. Ces indications devraient vous suffire pour accéder ici au paramètre de la ligne de commande (nous verrons précisément plus tard comment les paramètres sont gérés dans les programmes en C).

Erick Gallesio, S. Lavirotte
Polytech Nice Sophia
SI3

TD séances n° 3
Chaînes de caractères

6 Suppression d'une chaîne

Écrire la fonction

```
void suppression(char str[], const char suppr[]);
```

qui permet de supprimer la chaîne `suppr` dans la chaîne `str`:

```
Suppression de 'on' dans 'Bonjour' ⟶ 'Bjour'
Suppression de 'B' dans 'Bonjour' ⟶ 'onjour'
Suppression de 'Bonjour' dans 'Bonjour' ⟶ ''
Suppression de 'jour' dans 'Bonjour' ⟶ 'Bon'
Suppression de 'abc' dans 'Bonjour' ⟶ 'Bonjour'
```

Comme pour la fonction de suppression précédente, vous essaierez de ne pas utiliser de chaîne temporaire (la suppression se fait directement dans la chaîne `str`). Pour cela, vous pouvez vous définir une fonction qui décale une chaîne de caractères vers la gauche, à partir d'une position donnée, de `n` caractères.

7 Autres fonctions standard (facultatif)

En vous inspirant des pages de manuel correspondantes, implémenter les fonctions suivantes:

```
int strncmp(const char s1[], const char s2[], int n);
int strcasecmp(const char s1[], const char s2[]);
int strspn(const char s[], const char accept[]);
```

8 Mélange de caractères (facultatif)

Écrire la fonction

```
void shuffle(char str[]);
```

Cette fonction mélange le contenu de la chaîne `str` en utilisant le générateur de nombres aléatoires `rand`. Après l'exécution de cette fonction, `str` contient donc un anagramme de la chaîne passée en paramètre.

Voir les pages de manuel `random(3)` et, éventuellement, `srandom(3)`.

9 Mots en vrac dans une phrase (facultatif)

On sait que si l'on mélange les caractères au milieu d'un mot (c'est a dire si on laisse le premier caractère et le dernier caractère en place), ceui ci rest tout de même lisible s'il est dans une phrase (votre cerveau tentera de remettre les choses dans l'ordre. Par exemple, la phrase:

```
Quand la premiere lettre et la derniere lettre d'un mot restent en place,
on peut melanger les lettres du milieu sans trop gener la lecture.
```

devient

```
Qnaud la pmereire lrette et la dieernre lrette d'un mot renestt en plcae,
on puet meagenlr les lteetrs du miileu snas trop gneer la lteucre.
```

Ici, il y a pas mal de mélange, mais si on ne mélange qu'un mot sur deux, on obtient:

```
Quand la premiere lrette et la derniere lttere d'un mot resnett en place,
on peut menalegr les lrteets du mleiiu sans trop gener la lectrue.
```

Modifier la fonction `shuffle` précédente pour mélanger obtenir des phrases du type précédent.