Chaînes de caractères

Mardi 07 Novembre 2017

Michael FRANÇOIS

francois@esiea.fr

https://francois.esiea.fr/

Objectif de ce cours

- Comprendre les concepts de tableaux et chaînes de caractères.
- Savoir déclarer, remplir et manipuler des tableaux et des chaînes de caractères.

Tableaux de caractères

• Jusqu'à présent, on a utilisé des tableaux de caractères pour manipuler des ensembles de caractères.

• Par exemple pour déclarer et initialiser un tableau composé de 7 caractères ('B', 'o', 'n', 'j', 'o', 'u', 'r') :

```
char word[7] = {'B','o','n','j','o','u','r'};
```

• Une limite opérationnelle s'impose quand le nombre de caractères à manipuler explose.

Quelques problèmes :

- Si on ne connaît pas la taille du mot (ou de la phrase) que l'on souhaite mémoriser, quelle taille de tableau choisir ?
- Si le mot (ou la phrase) est d'une taille inférieure, comment gérer les cases non utilisées ?

```
char word[10] = {'B','o','n','j','o','u','r'};
```

Que valent les cases d'indices 7 à 9 ?

Quelques problèmes :

- Si on ne connaît pas la taille du mot (ou de la phrase) que l'on souhaite mémoriser, quelle taille de tableau choisir ?
- Si le mot (ou la phrase) est d'une taille inférieure, comment gérer les cases non utilisées ?

```
char word[10] = {'B','o','n','j','o','u','r'};
```

Que valent les cases d'indices 7 à 9 ?
Les cases de 7 à 9 contiennent le caractère `NUL` symbolisé par '\0' associé également à la valeur 0.

Comment s'en sortir ?

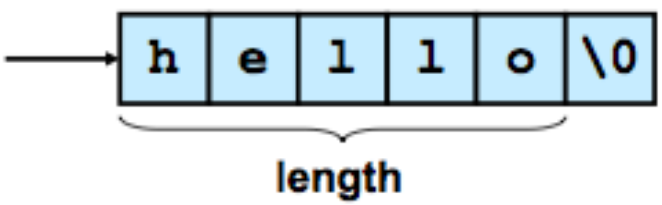
- prendre une taille (maximum) de tableau qui puisse convenir dans tous les cas ;
- ou mettre une balise spéciale de fin de partie utile dans le tableau : toutes les cases d'indice inférieur à celle contenant la balise contiennent la partie utile et toutes les autres contiennent de l'information non maîtrisée (état de la mémoire à l'instant du programme).

Chaînes de caractères

• ATTENTION : en C, il n'existe pas de variable de type chaîne.

• Par contre il existe une convention de représentation des chaînes qui consiste à placer un caractère de code NUL (\0) à la fin d'une succession d'octets représentant chacun des caractères de la chaîne.

• Ainsi, une chaîne de n caractères occupe en mémoire un emplacement de $n + 1$ octets. En mémoire, la chaîne "hello" est représentée ainsi :



- Une chaîne de caractères est tout simplement un tableau de caractères contenant une balise de fin. On peut donc utiliser l'opérateur [] pour accéder à une case en lecture ou écriture.
- La déclaration d'une chaîne de caractères suit le modèle suivant :

```
char word[10];
```

NB : `word` est un tableau de `char` dans lequel on peut stocker une chaîne d'au plus 9 caractères.

- Il est possible de déclarer et d'initialiser en même temps une chaîne de caractères en terminant par le caractère '\0'

```
char word[10] = {'B','O','N','J','O','U','R','\0'};
```

- En pratique, on utilise plutôt l'opérateur "" qui sépare la chaîne en caractères, fait la conversion caractère à caractère via la table ASCII et ajoute '\0' à la fin.

```
char word[10] = "BONJOUR";
```

Affichage d'une chaîne de caractères

Pour afficher une chaîne de caractère, on peut effectuer une boucle sur le tableau correspondant et afficher élément par élément la chaîne jusqu'au dernier caractère.

```c
#include <stdio.h>


int main()
{
  int i=0; char word[10] = "BONJOUR";

  while (word[i] != '\0')
  {
    printf("%c", word[i]);
    i++;
  }
  printf("\n");

  return 0;
}
```

```
BONJOUR
```

NB : cette façon marche mais n'est pas optimale puisqu'on peut effectuer cela en évitant la boucle.

Sachant que la chaîne se termine par '\0', il existe un moyen de l'afficher en une instruction en utilisant la fonction `printf` avec le descripteur `%s`, comme dans cet exemple :

```c
#include <stdio.h>


int main()
{
  char word[10] = "BONJOUR";

  printf("%s\n", word);

  return 0;
}
```

```
BONJOUR
```

NB : on a le même affichage en évitant les boucles et aussi les possibilités de faire des débordements de zone mémoire.

Comment saisir une chaîne de caractères depuis l'entrée standard ?

Il existe plusieurs fonctions qu'on peut utiliser pour récupérer une chaîne de caractères depuis l'entrée standard.

scanf()

• Pour saisir une chaîne de caractères, on peut utiliser la fonction `scanf` avec le descripteur `%s` et sans mettre `&` devant le nom du tableau :

```
scanf("%s", word);
```

• `scanf` remplira le tableau `word` lettre à lettre et mettra '\0' à la fin SAUF si la fonction rencontre un espace ou entrée.

Exemple : (Quand la chaîne saisie ne contient pas d'espace)

```c
#include <stdio.h>


int main()
{
  char word[20];

  printf("Saisir une chaîne de caractères : ");
  scanf("%s", word);
  printf("Chaîne saisie : %s\n", word);


  return 0;
}
```

Après exécution on obtient :

```
Saisir une chaîne de caractères : ESIEA
Chaîne saisie : ESIEA
```

Exemple : (Quand la chaîne saisie contient un espace)

```c
#include <stdio.h>


int main()
{
  char word[20];

  printf("Saisir une chaîne de caractères : ");
  scanf("%s", word);
  printf("Chaîne saisie : %s\n", word);


  return 0;
}
```

Après exécution on obtient :

```
Saisir une chaîne de caractères : Paris est magique !
Chaîne saisie : Paris
```

NB : la récupération des caractères s'est arrêtée juste avant l'espace.

Il est possible de spécialiser la fonction `scanf` en précisant entre [] la liste des caractères autorisés (donc tout autre caractère est interdit et provoque la fin du `scanf`)

Exemple : (on autorise que les caractères numériques)

```c
#include <stdio.h>


int main()
{
  char word[20];

  printf("Saisir une chaîne de caractères : ");
  scanf("%[0123456789]", word);
  printf("Chaîne saisie : %s\n", word);

  return 0;
}
```

Après exécution on obtient :

```
Saisir une chaîne de caractères : 123456789A
Chaîne saisie : 123456789
```

```
Saisir une chaîne de caractères : 12345F1234
Chaîne saisie : 12345
```

Il est possible également d'indiquer uniquement la liste des caractères qui sont interdits, en utilisant le symbole ^ . Cela permettrait de récupérer toute la chaîne saisie.

Exemple : (on autorise tous les caractères sauf **entrée**)

```c
#include <stdio.h>


int main()
{
  char word[20];

  printf("Saisir une chaîne de caractères : ");
  scanf("%[^\n]", word);
  printf("Chaîne saisie : %s\n", word);


  return 0;
}
```

Après exécution on obtient :

```
Saisir une chaîne de caractères : Paris est magique !
Chaîne saisie : Paris est magique !
```

NB : on voit ici qu'on a pu récupérer toute notre chaîne saisie y compris les espaces.

Il est également possible de limiter le nombre de caractères maximum à saisir. Cela permettrait de ne pas déborder sur l'espace mémoire réservé, car l'utilisateur peut saisir n'importe quoi en entrée.

Exemple : (on limite le nombre de caractères à saisir)

```c
#include <stdio.h>


int main()
{
  char word[20];

  printf("Saisir une chaîne de caractères : ");
  scanf("%19[^\n]", word);
  printf("Chaîne saisie : %s\n", word);


  return 0;
}
```

Après exécution on obtient :

```
Saisir une chaîne de caractères : Ici c'est Paris !!!!!!!!!!!!!
Chaîne saisie : Ici c'est Paris !!!
```

NB : on voit que seulement 19 caractères on été récupérés, comme ça à la 20 ème position il y aura le '\0' de fin de chaîne.

fgets()

• Cette fonction permet de lire une suite de caractères à partir d'un flux quelconque (l'entrée standard ou un fichier).

• Le nombre maximal de caractères à lire est de n-1 (où *n* désigne la taille du tableau), car le dernier caractère est réservé au zéro de fin de chaîne.

• Concernant l'adresse de retour, `fgets` fournit l'adresse de la chaîne lue, lorsque la lecture s'est bien déroulée. Elle renvoie le pointeur NULL en cas d'erreur.

Exemple : (récupération d'une chaîne avec `fgets`)

```c
#include <stdio.h>


int main()
{
  char word[20];

  printf("Saisir une chaîne de caractères : ");
  fgets(word, 20, stdin); /*19 caractères à lire au maximum*/
  printf("Chaîne saisie : %s\n", word);


  return 0;
}
```

Après exécution on obtient :

```
Saisir une chaîne de caractères : La MCN en force !!!!!!!
Chaîne saisie : La MCN en force !!!
```

NB : les caractères excédentaires restent disponibles dans le tampon pour une prochaine lecture.

Exemple : (récupération d'une chaîne avec son excédent)

```c
#include <stdio.h>


int main()
{
  char word[23];
  char excedent[10];

  printf("Saisir une chaîne de caractères : ");
  fgets(word, 23, stdin); /*22 caractères à lire au maximum*/
  printf("Chaîne saisie : %s\n", word);
  scanf("%9[^\n]", excedent); /*Pour récupérer le reste de la chaîne*/
  printf("Reste de la chaîne : %s\n", excedent);

  return 0;
}
```

Après exécution on obtient :

```
Saisir une chaîne de caractères : J'adore le langage C !YOUPI ...
Chaîne saisie : J'adore le langage C !
Reste de la chaîne : YOUPI ...
```

NB : on voit qu'on n'a pas eu à faire une deuxième saisie, car le `scanf` à récupéré directement les caractères restants lors de la première saisie.

La bibliothèque `string.h` contient de nombreuses fonctions utiles pour manipuler les chaînes comme :

- `strlen` qui renvoie le nombre de caractères utiles d'une chaîne de caractères.

- `strcat` qui concatène deux chaînes de caractères (*i.e.* met une chaîne de caractères à la suite d'une autre pour en faire une unique grande).

- `strcpy` qui copie une chaîne dans une autre.

- `strcmp` qui compare une chaîne de caractères avec une autre.

- etc.

Bibliographie

- L. BEAUDOIN, Introduction à l'algorithmique et au langage C (Chaînes de caractères), cours 1A 2016-2017 ESIEA-Paris.
- C. DELANNOY, Langage C, éditions EYROLLES, 4ème tirage 2005.