

IAP - TP n°4 - Gestion de chaînes de bas niveau

Objectif du TP

- Comprendre le fonctionnement des chaînes de caractères en C
- Manipulation des fonctions de traitement
- Écrire des fonctions en C avec le passage de paramètres

Remarque: Une grande partie du programme a été mise à jour après la séance. La dernière version du programme a été rendue avec ce compte rendu dans la zone de dépôt principale.

Le code à déboguer ayant lui été refait complètement, mais ne pouvant pas être inclus dans la zone de dépôt, il a été ajouté en annexe de ce compte rendu.

Partie 1 - Lecture et écriture d'une chaîne de caractères

Fonction lire_chaine et procédure ecrire_chaine:

```
---Traitements sur les chaines de caractères---
-1-Création d'une chaine---
-2-Affichage d'une chaine---
-3-Affectation de chaine---
-4-Comparaison de chaine---
-5-Longueur de chaine---
-6-Concaténation de chaine---
-7-Minuscule une chaine---
-8-Rechercher une sous-chaine par caractère---
-9-Rechercher une sous-chaine---
-10-Formater une chaine---
-0-Quitter---
---Choisissez une option (0-10)---
-->1

---Création de chaine---
Choisissez la chaine (1 ou 2)
1
Saisir la chaine:
-->blabla1

---Traitements sur les chaines de caractères---
-1-Création d'une chaine---
-2-Affichage d'une chaine---
-3-Affectation de chaine---
-4-Comparaison de chaine---
-5-Longueur de chaine---
-6-Concaténation de chaine---
-7-Minuscule une chaine---
-8-Rechercher une sous-chaine par caractère---
-9-Rechercher une sous-chaine---
-10-Formater une chaine---
-0-Quitter---
---Choisissez une option (0-10)---
-->2

---Affichage d'une chaine---
Choisissez la chaine (1 ou 2)
1
-->blabla1
```

lire_chaine() remplace
gets() et ecrire_chaine()
remplace puts().

Partie 2 - Fonctions de bases pour le traitement des chaînes de caractères

Fonction affectation_chaine :

```
---Création de chaine---
Choisissez la chaine (1 ou 2)
1
Saisir la chaine:
-->blabla

---Traitements sur les chaines de caractères---
-1-Création d'une chaine---
-2-Affichage d'une chaine---
-3-Affectation de chaine---
-4-Comparaison de chaine---
-5-Longueur de chaine---
-6-Concaténation de chaine---
-7-Minuscule une chaine---
-8-Rechercher une sous-chaine par caractère---
-9-Rechercher une sous-chaine---
-10-Formater une chaine---
-0-Quitter---
---Choisissez une option (0-10)---
-->3

---Affectation de chaine---
1: Ch1 <- Ch2
2: Ch2 <- Ch1
2

---Traitements sur les chaines de caractères---
-1-Création d'une chaine---
-2-Affichage d'une chaine---
-3-Affectation de chaine---
-4-Comparaison de chaine---
-5-Longueur de chaine---
-6-Concaténation de chaine---
-7-Minuscule une chaine---
-8-Rechercher une sous-chaine par caractère---
-9-Rechercher une sous-chaine---
-10-Formater une chaine---
-0-Quitter---
---Choisissez une option (0-10)---
-->2

---Affichage d'une chaine---
Choisissez la chaine (1 ou 2)
2
-->blabla
```

affectation_chaine() remplace strcpy().

Fonction `compare_chaine` :

`compare_chaine()` remplace `strcmp()`.

Comparaison de "ABC" et de "ABA":

```
---Comparaison de chaine---  
Résultat de la comparaison des deux chaines (ch1 <-> ch2) : 2
```

Comparaison de "ABA" et de "ABC":

```
---Comparaison de chaine---  
Résultat de la comparaison des deux chaines (ch1 <-> ch2) : -2
```

Comparaison de "ABC" et de "ABC":

```
---Comparaison de chaine---  
Résultat de la comparaison des deux chaines (ch1 <-> ch2) : 0
```

Fonction `longueur_chaine` :

```
---Création de chaine---  
Choisissez la chaine (1 ou 2)  
1  
Saisir la chaine:  
-->1234567  
  
---Traitements sur les chaines de caractères---  
-1-Création d'une chaine---  
-2-Affichage d'une chaine---  
-3-Affectation de chaine---  
-4-Comparaison de chaine---  
-5-Longueur de chaine---  
-6-Concaténation de chaine---  
-7-Minuscule une chaine---  
-8-Rechercher une sous-chaine par caractère---  
-9-Rechercher une sous-chaine---  
-10-Formater une chaine---  
-0-Quitter---  
---Choisissez une option (0-10)---  
-->5  
  
---Longueur de chaine---  
Choisir la chaine (1 ou 2)  
1  
Longueur Ch1 : 7
```

`longueur_chaine()` remplace
`strlen()`.

Fonction concatene_chaine :

```
-->6

---Concaténation de chaine---
1: Ch1 = Ch1 + Ch2
2: Ch2 = Ch2 + Ch1
1

---Traitements sur les chaines de caractères---
-1-Création d'une chaine---
-2-Affichage d'une chaine---
-3-Affectation de chaine---
-4-Comparaison de chaine---
-5-Longueur de chaine---
-6-Concaténation de chaine---
-7-Minuscule une chaine---
-8-Rechercher une sous-chaine par caractère---
-9-Rechercher une sous-chaine---
-10-Formater une chaine---
-0-Quitter---
---Choisissez une option (0-10)---
-->2

---Affichage d'une chaine---
Choisissez la chaine (1 ou 2)
1
-->ABC123
```

Concaténation de la chaîne 1
("ABC") et de la chaîne 2 ("123").
concatene_chaine() remplace
strcat()

Partie III - Fonctions de traitements avancés sur les chaînes

Fonction min_chaine :

```
---Minuscule une chaine---
Choisir la chaine (1 ou 2)
1
test1

---Traitements sur les chaines de caractères---
-1-Création d'une chaine---
-2-Affichage d'une chaine---
-3-Affectation de chaine---
-4-Comparaison de chaine---
-5-Longueur de chaine---
-6-Concaténation de chaine---
-7-Minuscule une chaine---
-8-Rechercher une sous-chaine par caractère---
-9-Rechercher une sous-chaine---
-10-Formater une chaine---
-0-Quitter---
---Choisissez une option (0-10)---
-->7

---Minuscule une chaine---
Choisir la chaine (1 ou 2)
2
test2
```

Minuscule de la chaîne 1 ("TEST1") et de la chaîne 2 ("Test2").

min_chaine() ne remplace aucune fonction de string.h, mais fonctionne comme la fonction tolower() de ctype.h (tolower() ne fonctionne que pour un seul caractère).

Fonction trouve_car_chaine et trouve_sschaine:

```
---Rechercher une sous-chaine par caractère---
Choisir la chaine (1 ou 2)
1
Saisir le caractère à rechercher:
D
DEFG

---Traitements sur les chaines de caractères---
-1-Création d'une chaine---
-2-Affichage d'une chaine---
-3-Affectation de chaine---
-4-Comparaison de chaine---
-5-Longueur de chaine---
-6-Concaténation de chaine---
-7-Minuscule une chaine---
-8-Rechercher une sous-chaine par caractère---
-9-Rechercher une sous-chaine---
-10-Formater une chaine---
-0-Quitter---
---Choisissez une option (0-10)---
-->9

---Rechercher une sous-chaine---
Choisir la chaine (1 ou 2)
1
Saisir la sous-chaine à rechercher:
EFG
La position de la sous-chaine dans la chaine 1 est 4.
```

Travail des deux fonctions sur la chaîne 1 ("ABCDEFGF").

trouve_car_chaine() remplace strchr().

trouve_sschaine() remplace strstr().

Partie IV - Déboguer le traitement consistant à formater une chaîne de caractères

n°	Ligne	Erreur	Solution
1	11	MAXCARS mal défini	#define MAXCARS 80
2	13	Point virgule manquant	Ajout du point virgule
3	26	Crochet en trop	Suppression du crochet
4	32	'&' manquant dans scanf	scanf("%c", &rep);
5	33	Entrée standard non vidée	Ajout de getchar();
6	34	Problème de condition	!= au lieu de =!
7	53	Point virgule en trop	Suppression du point virgule
8	55	ch2 n'est pas un pointeur	char * ch2;
9	58	state n'a pas de valeur	int state = 0;
10	78	Égal en trop	ch[i] = ch2[j];
11	88	Pas d'incrément de j	j++;

Code à déboguer:

```
/******B2******/
/* NOM : AMICHI      PRENOM : Lounès      Groupe TP : B2      */
/* DATE : 03/12/25      */
/* FICHER : formate_a_deboguer.c      */
/* DESCRIPTION : Traitement de formatage d'une chaine de caracteres */
/*******/
#include <stdio.h>
#include <stdlib.h>
#include <string.h>

/* Correction 1: rajout du 0 manquant de MAXCARS */
#define MAXCARS 80
/* Correction 2: Ajout du point-virgule à la fin du prototype */
char* formater_chaine(char ch[]);

int main()
{
    char chaine_a_formater[MAXCARS+1];
    char rep;
    do
    {
        printf("Saisir une chaine mal formatee pouvant avoir plusieurs espaces et tabulations \n");
        printf(" en début, en fin et entres les mots : ");
        fgets(chaine_a_formater,MAXCARS+1,stdin);
        /* Correction 3: Suppression du crochet ']' en trop */
        chaine_a_formater[strlen(chaine_a_formater)-1]='\0';
        printf("Chaine saisie : '%s' \n",chaine_a_formater);
        formater_chaine(chaine_a_formater);
        printf("Chaine formatee : '%s' \n",chaine_a_formater);
        printf("Voulez vous recommencer (o/n):");
        /* Correction 4: Ajout du '&' */
        scanf("%c", &rep);
        /* Correction 5: Suppression du RC (le \n reste sinon et bloque le prochain fgets) */
        getchar();
    }
    /* Correction 6: != */
    while ((rep!='o') && (rep!='O'));

    printf("FIN PROGRAMME FORMATER CHAINE !!! \n");
    return EXIT_SUCCESS;
}

/*******/
/* NOM FONCTION : formater_chaine      */
/* DESCRIPTION : Realise le formatage d'une chaine en supprimant */
/* les extra caracteres qu'elles contient aux extremités ou entre les mots */
/* Retourne :un pointeur sur la chaine formater      */
/* Effets de bords : Saisie de la chaine a formatee au clavier      */
/* Paramètres en entree : ch (la chaine a formater)      */
/* Paramètres en sortie : ch (la chaine formatee)      */
/*******/
```

```

/* Correction 7: Suppression du point-virgule qui empêchait la définition de la fonction */
char* formater_chaine(char ch[])
{
    /* Correction 8: ch2 doit être un pointeur pour faire ch2=ch.*/
    char * ch2; // Sert a parcourir la chaine a formater
    int i=0; // Compteur pour parcourir ch le tableau ou l'on sauvegarde la chaine formatee
    int j=0; // Compteur utiliser pour parcourir ch2 le tableau correspondant a la chaine a formater
    /* Correction 9: Initialisation de state à 0 */
    int state=0; // Defini l'état courant de l'automate a etats finis
    ch2=ch;
    while(ch2[j])
    {
        switch (state)
        {
            case 0 : // Hors mot
                if ((ch2[j]!=' ') && (ch2[j]!='\t') && (ch2[j]!=13) && (ch2[j]!=10))
                {
                    state=1;
                    ch[i]=ch2[j];
                    i++;
                }
                break;

            case 1 :// Dans mot
                if ((ch2[j]!=' ') && (ch2[j]!='\t') && (ch2[j]!=13) && (ch2[j]!=10))
                {
                    /* Correction 10: suppression d'un = */
                    ch[i]=ch2[j];
                    i++;
                }
                else {
                    ch[i]=' ';
                    i++;
                    state=0;
                }
                break;
            default : printf("Cas impossible !!!\n\n");
        }

        /* Correction 11: Incrémentation de j pour avancer dans la boucle */
        j++;
    }

    //Traitement de fin !!!
    if ((state==0) && (i>0) && (ch[i-1]!=' '))
        ch[i-1]='\0';
    else ch[i]='\0';

    return ch;
}

```