Université Mohamed Premier
École Supérieure de Technologie
Oujda

Filière Génie Civil - Semestre 2
Année universitaire : $2017 - 2018$
Module : Informatique (**Langage** C)

Solutions des TD/TP N° 2

Objectifs d'apprentissage

- Maîtriser la manipulation des chaînes de caractères.

Exercice 1 (TD)

On suppose que **ch** est une variable qui représente une chaîne de caractères d'au plus 20 caractères.

1. L'instruction qui permet de déclarer la variable **ch** sans l'initialiser :

   ```
   char ch[21];
   ```

2. L'instruction qui permet de déclarer la variable **ch** et de l'initialiser avec la chaîne "Programmation en C".

   ```
   char ch[21] = "Programmation en C";
   ```

3. L'instruction qui permet d'afficher le contenu de la variable **ch** :

   ```
   /* Solution avec printf */

   printf("%s", ch);

   /* Solution avec puts */

   puts(ch);
   ```

4. L'instruction qui permet d'afficher la longueur de chaîne **ch** :

   ```
   #include <string.h>
   /* inclure <string.h> quelque part en haut */

   printf("%d", strlen(ch));
   ```

5. L'instruction qui permet d'afficher respectivement, le premier et le dernier caractère de **ch** :

   ```
   printf("%c%c", ch[0], ch[strlen(ch) - 1]);
   ```

6. L'instruction qui permet d'effacer tout le contenu de la variable **ch** (deux solutions possibles) :

```
/* Première solution : copier la chaîne vide dans ch */

strcpy(ch, "");

/* Deuxième solution : mettre '\0' dans ch[0] */

ch[0] = '\0';
```

7. L'instruction qui permet de lire une suite de caractères et de la ranger dans la variable **ch** :

```
/* Première solution avec scanf : ignore les caractères
   après le premier caractère blanc saisi, et ne
   permet pas de lire la chaîne vide */

scanf("%s", ch);

/* Deuxième solution avec gets : efficace */

gets(ch);
```

Exercice 2 (TD)

Soit le programme :

```
main()
{
   char s[10];

   strcpy(s, "abc");
   printf("%d %d", strlen(s), sizeof(s));
   system("pause");
}
```

La bonne réponse est (A), et le programme affiche :

```
3 10
```

Justification :

- L'instruction **strcpy**(s, "abc") copie la chaîne "abc" dans s.

– **strlen**(s) : donne la longueur de la chaîne s, c'est-à-dire le nombre de caractères effectivement stockés dans s. C'est donc 3.

– **sizeof**(s) : donne la taille en octets du tableau s, qui est égale à 10.

Exercice 3 (TD)

Écrire un programme qui détermine le nombre de lettres e (minuscules) présentes dans un texte de moins d'une ligne (supposée ne pas dépasser 132 caractères) fourni au clavier.

```
#include <stdio.h>
#include <string.h>
#define N  132

main()
{
   char texte[N + 1]; /* chaîne de caractères pour stoker le texte */
   int i;             /* compteur de la boucle for */
   int Nbe = 0;       /* Nombre de lettres e dans le texte */
   int lg;            /* la longueur effective du texte */

   printf("Saisir votre texte (Maximum 132 caracteres) : ");
   gets(texte);
   lg = strlen(texte);
   for(i = 0; i < lg; i++)
   {
      if(texte[i] == 'e')
        Nbe++;
   }
   printf("Ce texte contient %d lettres e\n", Nbe);
   system("pause");
}
```

Une deuxième solution possible sans utiliser la fonction **strlen** :

```
#include <stdio.h>
#include <string.h>
#define N  132

main()
{
   char texte[N + 1]; /* chaîne de caractères pour stoker le texte */
```

```
   int i;             /* compteur de la boucle for */
   int Nbe = 0;       /* Nombre de lettres e dans le texte */

   printf("Saisir votre texte (Maximum 132 caracteres) : ");
   gets(texte);
   for(i = 0; texte[i] != '\0'; i++)
   {
      if(texte[i] == 'e')
        Nbe++;
   }
   printf("Ce texte contient %d lettres e\n", Nbe);
   system("pause");
}
```

Une version de cette solution en utilisant la boucle **while** :

```
#include <stdio.h>
#include <string.h>
#define N  132

main()
{
   char texte[N + 1]; /* chaîne de caractères pour stoker le texte */
   int i;
   int Nbe = 0;       /* Nombre de lettres e dans le texte */

   printf("Saisir votre texte (Maximum 132 caracteres) : ");
   gets(texte);
   i = 0;
   while(texte[i] != '\0')
   {
      if(texte[i] == 'e')
        Nbe++;
      i++;
   }
   printf("Ce texte contient %d lettres e\n", Nbe);
   system("pause");
}
```

Exercice 4 (TD)

Écrire un programme qui détermine si un mot (d'au plus 30 caractères) est un palindrome, c'est-à-dire un mot qui se lit de la même façon dans les deux sens

(exemples : ailia, non).

```
#include <stdio.h>
#include <string.h>
#define N  30

main()
{
   char mot[N + 1]; /* chaîne de caractères pour stoker le mot */
   int i;           /* compteur */
   int L;           /* longueur du mot */

   printf("Saisir un mot S.V.P (Maximum 30 caracteres) : ");
   gets(mot);
   L = strlen(mot);
   if(L <= 1)
     printf("Le mot %s est un palindrome\n", mot);
   else
   {
      for(i = 0; i < L / 2; i++)
      {
         if(mot[i] != mot[L - 1 - i])
         {
            printf("Le mot %s n'est pas un palindrome\n", mot);
            system("pause");
            exit(0);
         }
      }
      printf("Le mot %s est un palindrome\n", mot);
   }
   system("pause");
}
```

Une autre solution avec la boucle **while** :

```
#include <stdio.h>
#include <string.h>
#define N  30

main()
{
   char mot[N + 1]; /* chaîne de caractères pour stoker le mot */
   int i, j;        /* compteurs */
```

```
   int L;           /* longueur du mot */

   printf("Saisir un mot S.V.P (Maximum 30 caracteres) : ");
   gets(mot);
   L = strlen(mot);
   i = 0;
   j = L - 1;
   while(i < j)
   {
      if(mot[i] != mot[j])
      {
         printf("Le mot %s n'est pas un palindrome\n", mot);
         system("pause");
         exit(0);
      }
      else
      {
         i++;
         j--;
      }
   }
   printf("Le mot %s est un palindrome\n", mot);
   system("pause");
}
```

Exercice 5 (TP)

Écrire un programme qui lit un verbe du premier groupe et qui en affiche la conjugaison au présent de l'indicatif, sous la forme :

```
je chante
tu chantes
il chante
nous chantons
vous chantez
ils chantent
```

Le programme devra vérifier que le mot fourni se termine bien par "er". On supposera qu'il ne peut comporter plus de 26 lettres et qu'il s'agit d'un verbe régulier. Autrement dit, on admettra que l'utilisateur ne fournira pas un verbe tel que "manger" (le programme afficherait alors : "nous mangons" !).

```
#include <stdio.h>
```

```
#include <string.h>
#define N  26

main()
{
   char verbe[N + 1];   /* chaîne de caractères pour stoker le verbe */
   char radical[N - 1] = "";
                        /* chaîne de caractères pour stoker le radical
                           du verbe */
   int L;               /* longueur du mot */

   printf("Saisir un verbe du premier groupe (Maximum 26 caracteres) : ");
   gets(verbe);
   L = strlen(verbe);
   /* on vérifie si le verbe se termine par "er" */
   if(verbe[L - 2] != 'e' || verbe[L - 1] != 'r')
     printf("Desole, le verbe ne se termine pas par er\n");
   else
   {
      printf("Merci, le verbe se termine par er\n");
      strncpy(radical, verbe, L - 2);
      printf("Voici sa conjugaison au present de l'indicatif :\n");
      printf("je %se\n", radical);
      printf("tu %ses\n", radical);
      printf("il %se\n", radical);
      printf("elle %se\n", radical);
      printf("nous %sons\n", radical);
      printf("vous %sez\n", radical);
      printf("ils %sent\n", radical);
      printf("elles %sent\n", radical);
   }
   system("pause");
}
```