Langage de programmation C :

Les Chaînes de Caractères

1) Définition et Déclarations

- Les variables de types char ne peuvent recevoir qu'un seul caractère.

- Une chaîne de caractères est une suite de caractères alphanumériques (du texte) qui se termine par le caractère nul \0.
- Elle est représenté sur une suite d'octets se terminant par un octet supplémentaire lié au symbole '\0'. Celui-ci indique une fin de chaîne.
- Une chaîne de caractères est considérée comme un tableau de caractères.
- Il n'existe pas de type de données pour les chaînes de caractères.

Déclarations

- **Sous forme de tableau**

  char <*nomchaine*> [<longueur>];

- **Sous forme de pointeur**

  Char *<*nomchaine*>;

- **Exemples**

  Char nom[20];

  Char *prenom;

2) Chaînes de caractères constantes(chaînes littérales)

- Sont représentées entre guillemets.
  - La chaîne vide est noté ''''.
- Dans une chaîne, les caractères de contrôle peuvent être utilisés.

- **Exemples:**
  - ''ce \n texte \n sera réparti sur 3 lignes.''
  - ''Affichage de \'' guillemets \'' \n''
  - ''un'' ''deux'' ''trois'' sera évaluée comme ''un deux trois''

3)Tableaux de caractères

- Pour stocker une chaîne de six caractères, il faut déclarer un tableau de type char avec sept éléments:
  - char chaine[7];
  - La septième case est réservée pour le caractère nul \0, même s'il est représenté par deux caractères, il est interprété comme un seul caractère et sa valeur ASCII est 0.
- Initialiser le tableau des caractères
  - Char chaine[10]={'b','o','n','j','o','u','r','\0'};
  - Char chaine[10]=''bonjour'';
  - Char chaine[]=''bonjour'';
  - Char chaine[7]=''bonjour''; */*erreur pendant l'exécution*/*
  - Char chaine[6]=''bonjour''; */*erreur pendant la compilation*/*

4) Chaînes et pointeurs

- Une chaîne de caractères est stockée dans un tableau de type char, et la fin de cette chaîne est représentée par le caractère nul. Pour définir une chaîne, il suffit donc de pointer au début de cette chaîne. On peut donc représenté le début de la chaîne par un pointeur vers une variable char :
  - **char *message**;

  Cette instruction déclare le pointeur message vers une variable de type char, mais le pointeur ne pointe encore sur rien. Si on écrit:
  - **char *message=''bonjour tout le monde'';**

  La chaîne sera stockée quelque part dans la mémoire(avec

  un caractère nul à la fin).

  Voici une instruction équivalente à la précédente:
  - **char message[]=''bonjour tout le monde '';**

- **La fonction malloc:**
  - La fonction **malloc()** est une fonction qui permet de réserver de l'espace mémoire. On lui transmet en argument le nombre d'octets nécessaires, elle se charge de trouver et de réserver un bloc de mémoire libre.
  - Exemple :
    - char *p; p=malloc(1); *p='a';
    - char *pt; pt=malloc(100);
- Remarques:
  - char *ch1=''une chaîne''; char *ch2=''une autre chaîne'';
  - ch1=ch2; /* *ch1 et ch2 pointent sur la même chaîne*/*
  - char ch1[20]=''une chaîne'';
  - char ch2[20]=''une autre chaîne'';
  - char ch3[30];
  - ch1=ch2; /* *impossible affichage d'erreur*/*
  - ch3= ''bonjour''; /* *impossible affichage d'erreur*/*

5)Ordre alphabétique et lexicographique

- Ordre alphabétique des caractères:
  - Pour le code ASCII, on a l'ordre suivant :
    - …,0,1,2,…,9,…,A,B,…Z, ,b,…,c
  - Il s'agit d'une relation d'ordre 'est inférieur à' sur l'ensemble des caractères.
  - Exemple:
    - '0' est inférieur à 'Z' et noté : '0'< 'Z'[(ASCII('0'=48), ASCII('Z'=90)]
- Ordre lexicographique des chaînes de caractères:
  - Il se base sur l'ordre alphabétique des caractères.
  - La chaîne vide '''' précède toutes les autres chaînes.
  - Exemples:
    - ''ABC'' précède ''BCD'' car 'A' <'B'
    - ''Abc'' précède ''abc''
    - ''ab'' précède ''abcd'' car '' '' précède ''cd''
    - '' ab'' précède ''ab'' car [(ASCII(' '=12), ASCII('a'=97)]

6)Manipulation des chaînes de caractères

- Fonction de <stdio.h>
  - Affichage de chaînes de caractères
    - char ch[]=''bonjour tout le monde'';
    - printf(''%s'',ch);
    - char *ch1;
    - puts(ch1); équivalente à printf(''%s'',ch1);
  - Lecture de chaînes de caractères
    - char lieu[30];
    - printf(''entrez le lieu de naissance'');scanf(''%s'',lieu);
    - char chaine[80];
    - printf(''entrez une chaine''); gets(chaine);
    - Contrairement à scanf, la fonction gets permet de saisir des chaînes de caractères contenant des espaces et des tabulations.

Quelques Fonctions de <string.h>

- Fonctions de <string.h>
  - **Longueur d'une chaîne de caractères**
    - La fonction **strlen** calcule la longueur d'une chaîne de caractères en octets, le caractère nul n'étant pas compté.
    - le nombre de caractères comptés est retourné au programme par la fonction, cela sous forme de valeur entière.
    - *Exemple :*
      - c=**strlen**("bonjour");
      - c aura la valeur 7.
  - **Concaténation de chaînes de caractères**
    - On peut concaténer deux chaînes de caractères en accrochant l'une d'elles à la fin de l'autre. Cette opération s'effectue via la fonction **strcat.**
    - *Exemple:*
      - **strcat**(ch1,ch2);

- Fonctions de <string.h>
  - **Comparaison de chaînes de caractères**
    - La fonction **strcmp** compare deux chaînes caractère par caractère, jusqu'à ce que soit détectée une différence ou que soit atteint le caractère nul.
    - La comparaison s'effectue dans l'ordre alphanumérique. En d'autres termes, on vérifie à chaque fois si les deux caractères à comparer occupent ou non la même place dans la table des caractères utilisés.
    - Ici, un caractère est considéré comme supérieur à un autre s'il occupe une place plus élevée dans la table des caractères, et inférieur s'il occupe une place moins élevée. Ainsi, dans la table ASCII le caractère 'Z' est plus grand que 'A' mais plus petit que 'a'.

- Fonctions de <string.h>
  - **Comparaison de chaînes de caractères**
    - Cette fonction retourne une valeur entière avec les conventions suivantes :
      - si la valeur est inférieure à 0, alors la chaîne dont l'adresse est donnée par le premier paramètre de la fonction est inférieure à l'autre chaîne
      - si la valeur est égale à 0, alors les deux chaînes sont égales
      - si la valeur est supérieure à 0, alors la chaîne dont l'adresse est donnée par le premier paramètre de la fonction est supérieure à l'autre chaîne.
      - La valeur transmise au programme peut être stockée dans une variable de type int.
    - Exemple:
      - int result;
      - result=strcmp(ch1,ch2);

- Fonctions de <string.h>
  - **Copie de chaînes de caractères**
    - La fonction ***strcpy()*** (prononcez *string copy*) est une fonction qui permet de copier une chaîne entière de caractères dans une autre. Cette fonction admet comme paramètres les deux chaînes de caractères. Elle retourne 1 si la copie s'est effectuée correctement, sinon elle renvoie 0.
    - Exemple:
      - **strcopy**(ch1,ch2);
    - **strncpy**, est similaire à strcopy, on notera que:
      - si la chaîne source a moins de n caractères non nuls, strncpy rajoutera n - strlen(source) caractères nuls à la suite, pour compléter;
      - si la chaîne source fait au moins n caractères, la fonction n'insèrera pas de caractère nul en fin de chaine (i.e. destination ne sera pas une chaîne de caractères valide).

- Fonctions de <string.h>
  - **Copie de chaînes de caractères**
    - La fonction ***strcpy()*** (prononcez *string copy*) est une fonction qui permet de copier une chaîne entière de caractères dans une autre. Cette fonction admet comme paramètres les deux chaînes de caractères. Elle retourne 1 si la copie s'est effectuée correctement, sinon elle renvoie 0.
    - Exemple:
      - **strcopy**(ch1,ch2);
    - **strncpy**(ch1,ch2,n), est similaire à strcopy, on notera que:
      - si la chaîne source a moins de n caractères non nuls, strncpy rajoutera n - strlen(source) caractères nuls à la suite, pour compléter;
      - si la chaîne source fait au moins n caractères, la fonction n'insèrera pas de caractère nul en fin de chaine (i.e. destination ne sera pas une chaîne de caractères valide).

- Fonctions de <string.h>
  - **Recherche dans une chaîne**
    - Les fonctions strchr & strrchr recherche le *caractère* dans la *chaîne* et renvoie la position de la première occurrence dans le cas de strchr et la position de la dernière occurrence dans le cas de strrchr.
    - **Exemple :**

```
#include <stdio.h>
#include <string.h>
 main() {
char  *str="E.N.A.C.", c = '.',  *ptr;
if ( (ptr = strrchr(str, c)) != NULL)
      printf("Le caractere %c est en position %d\n", c, ptr-str);
else
     printf("Pas trouve\n"); }
```

```
/* résultat de l'exécution */
 Le caractere  .  est en position 6
```

Manipulation des chaînes de caractères

```
#include <stdio.h>
#include <string.h>
 main() {
char *str = "E.N.A.C.", c = 'A', *ptr;
if ( (ptr = strchr(str, c)) != NULL )

printf("Le caractere %c est a la position %d\n", c, ptr-str);
else
        printf("Le caractere %c n'est pas trouve dans %s\n", c, str)
}
/* résultat de l'exécution*/
 Le caractere A est a la position 4
```

Manipulation des chaînes de caractères

```
#include <stdio.h>
#include <string.h>
 main() {
char str[10];
 char *ptr = "ABCDEFGHIJ";
strcpy(str, ptr); printf("%s\n", str);
} /* résultat de l'exécution: ABCDEFGHIJ*/
#include <stdio.h>
#include <string.h>
 main() {
      char str[10];
  char *ptr = "ABCDEFGHI";
strncpy(str, ptr, 3);
str[3] = '\0';
 printf("%s\n", str); } /* résultat de l'exécution: ABC */
```

Manipulation des chaînes de caractères

```
#include <stdio.h>
#include <string.h>
main() {
char *str = "E.N.A.C.";
printf("%d\n", strlen(str));} /* résultat de l'exécution: 8 */

#include <string.h>
 #include <stdio.h>
main() {
char m1[60] = «Brevet de Technicien Supérieur ";
char *m2 = «DWFS";
strcat(m1, m2); printf("%s\n", m1); }

/* résultat de l'exécution Brevet de Technicien Supérieur DWFS */
```