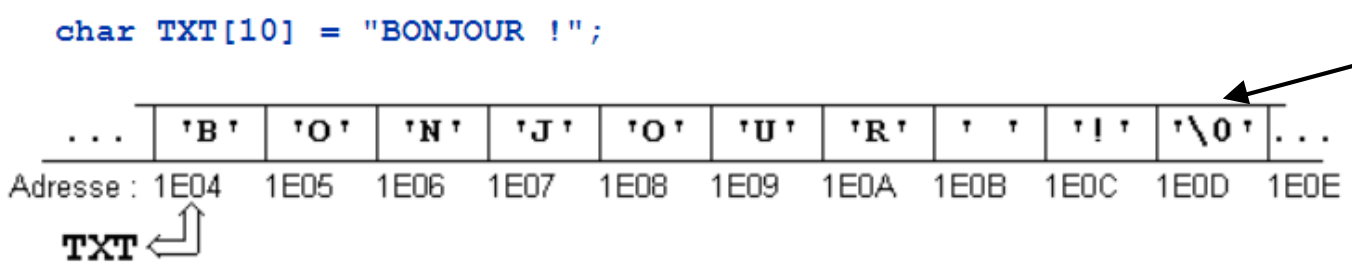
LES CHAINES DE CARACTÈRES

Algorithmique et programmation

Introduction

- Une chaîne de caractères (String en anglais) est un tableau d'éléments de type char, dont le dernier élément est le caractère nul \0.
- Le caractère \0 indique la fin de la chaîne de caractères.
- Une chaîne composée de n éléments sera en fait un tableau de n+1 éléments de type char.
- Déclarer une chaîne est aussi simple que déclarer un tableau à une dimension. Vous trouverez ci-dessous la syntaxe de base pour déclarer une chaîne.
  - *char nom_chaine[taille];*



Caractère de fin de chaine

TXT est un pointeur vers le premier caractère de la chaine

taille = 10 mais le nombre de caractères affichables est seulement 7

Initialisation

- char ville[] = "Meknes";
- char ville[20] = "Meknes";
- char ville[]={'M','e','k','n','e','s','\0'};
- char ville[20]={'M','e','k','n','e','s','\0’};
- char *ville = "Meknes";

```
char TXT[] = "Hello";
```

TXT:	'H'	'e'	'l'	'l'	'o'	'\0'

```
char TXT[6] = "Hello";
```

TXT:	'H'	'e'	'l'	'l'	'o'	'\0'

```
char TXT[8] = "Hello";
```

TXT:	'H'	'e'	'l'	'l'	'o'	'\0'	0	0

```
char TXT[5] = "Hello";
```

TXT:	'H'	'e'	'l'	'l'	'o'	☠

ERREUR pendant l'exécution

```
char TXT[4] = "Hello";
```

ERREUR pendant la compilation

Déclaration statique – saisie et affichage

```c
#include<stdio.h>
#include<stdlib.h>
main()
{
    char ville[20];
    printf("saisir votre ville\n");
    scanf("%s", ville);
    printf("%s", ville);

    system("pause");
}
```

Pas de &

Format spécifiant des chaines de caractères

Déclaration dynamique – saisie et affichage

```c
#include<stdio.h>
#include<stdlib.h>
main()
{
    char* ville;
    int taille = 20;
    ville = (char*)malloc(taille*sizeof(char));
    if(ville==NULL)exit(-1);
    //lecture
    printf("entrer votre ville : ");
    gets(ville);
    //affichage
    printf("votre ville est : ");
    puts(ville);
    free(ville);
    system("pause");
}
```

On peut utiliser gets au lieu de scanf("% s",ville)

On peut utiliser puts au lieu de printf("% s",ville)

Exercices – part1

- **Exercice 1: Calculer la longueur d'une chaîne de caractères**
- **Exercice 2: Calculer le nombre d'apparition (d'occurrence) d'un caractère dans une chaîne de caractères.**

Une fonction qui retourne une chaine

```c
#include<stdio.h>
#include<stdlib.h>
char *getVille()
{
    char *ville  =  "Meknes";
    return ville;
}
main(void)
{
    printf("%s", getVille());
    system("pause");
}
```

Chaine comme paramètre d'une fonction

```c
#include<stdio.h>
#include<stdlib.h>
void afficher(char str[])
{
    printf("%s ", str);
}

main(void)
{
    char ville[20];
    scanf("%s",ville);

    afficher(ville);
        system("pause");
}
```

Chaine comme paramètre d'une fonction

```c
#include< stdio.h>
char *getVille()
{
    char ville[]  =  "Meknes";
    return ville;
}
int main(void){
    printf("%s", getVille());
    return 0;
}
```

```
prog.c:7:12: warning: address of stack memory associated with local variable 'ville' returned [-Wreturn-stack-address]
return ville;
^~~~~
1 warning generated.
```

Les fonctions de traitement

- Pour pouvoir utiliser les fonctions de traitement sur les chaines de caractères nous devons inclure la bibliothèque <string.h>.

Soit : char *txt = " une 1ere chaine "; char *txt2 = " une 2eme chaine ";

- **strlen(txt) :** fournit la longueur de la chaîne ***sans*** compter le '\0' final
- **strcpy(txt , txt2) :** copie **txt2** vers **txt**
- **strcat(txt , txt2) :** ajoute **txt2** à la fin de **txt**
- **strcmp(txt , txt2):** compare **txt** et **txt2** lexicographiquement et fournit un résultat:
  - *Négatif si **txt** précède **txt2***
  - *Zéro si **txt** est égal à **txt2***
  - *Positif si **txt** suit **txt2***
- **strncpy(txt, txt2, n)** copie au plus **n** caractères de **txt2** vers **txt**
- **strncat(txt, txt2, n)** ajoute au plus **n** caractères de **txt2** à la fin de **txt**

Précédence lexicographique

- En nous basant sur cette relation de *précédence alphabétique des caractères*, nous pouvons définir une ***précédence lexicographique*** *pour les chaînes de caractères*. Cette relation de précédence suit l'<<ordre du dictionnaire>> et est définie de façon récurrente:
- a) La chaîne vide **""** précède lexicographiquement toutes les autres chaînes.
- b) La chaîne A = "a_1a_2a ... a_p" (p caractères) précède lexicographiquement la chaîne B = "b_1b_2 ... b_m" (m caractères) si l'une des deux conditions suivantes est remplie:
- 1) 'a_1' < 'b_1'
- 2) 'a_1' = 'b_1' et "a_2a_3 ... a_p" précède lexicographiquement "b_2b_3 ... b_m "
- Exemple :
  - *"ABC" précède "BCD" car 'A'<'B'*
  - *"ABC" précède "B" car 'A'<'B'*
  - *"Abc" précède "abc" car 'A'<'a'*
  - *"ab" précède "abcd" car "" précède "cd"*
  - *" ab" précède "ab" car ' '<'a' (le code ASCII de ' ' est 32, et le code ASCII de 'a' est 97)*

Les codes ASCII

- **Caractère de commandes** (0–31 & 127) : les caractères de contrôle sont des caractères non imprimables. Ils sont utilisés pour transmettre des commandes au PC ou à l'imprimante et sont basés sur la technologie de télécopie. Ces caractères sont utilisés, par exemple, pour définir des sauts de ligne ou des tabulations. Bon nombre de ces caractères ne sont plus utilisés aujourd'hui.
- **Caractères spéciaux** (32–47 / 58–64 / 91–96 / 123–126) : les caractères spéciaux comprennent tous les caractères imprimables qui ne sont ni des lettres ni des chiffres, tels que les signes de ponctuation ou les caractères techniques et mathématiques. ASCII inclut également le caractère blanc, qui est considéré comme un caractère non visible mais imprimable, et n'appartient donc pas aux caractères de contrôle comme on pourrait le soupçonner.
- **Nombres** (30–39) : les nombres comprennent les dix chiffres arabes de zéro à neuf.
- **Lettres** (65–90 / 97–122) : les lettres sont divisées en deux blocs, le premier groupe contenant les majuscules et le second les minuscules.
  - *https://www.ionos.fr/digitalguide/serveur/know-how/ascii-american-standard-code-for-information-interchange/*

Exemple d'utilisation

```c
#include<string.h>
main()
{
    char txt[7] = "Hello ";
    char txt2[7] = "world!";
    printf("la longuer de la chaine txt2 est: %d\n", strlen(txt2));
    //concaténation des deux chaines
    strcat(txt, txt2);
    printf("la chaine txt apres concatenation est: %s\n", txt);
    //copie d'une chaine dans une autre
    strcpy(txt2, "Le monde");
    printf("la chaine txt2 apres copie est:  %s\n", txt2);
    //comparaison de deux chaines de caratères
    if(strcmp(txt, txt2)==0){
        printf("les deux chaines txt et txt2 sont egaux\n");
    }else if(strcmp(txt, txt2)<0){
        printf("la chaine txt precede txt2\n");
    }else {
        printf("la chaine txt suit txt2\n");
    }
    strncpy(txt, txt2, 5);
    printf("la chaine txt apres copie de n caractres depuis txt2 est:  %s\n", txt);
    system("pause");
}
```

Inclusion de string,h

Fonctions de conversion

- La bibliothèque <***stdlib***> contient des déclarations de fonctions pour la conversion de nombres en chaînes de caractères et vice-versa.
- **atoi**: une fonction qui convertit une chaîne de caractères en une valeur numérique (valeur de retour) de type **int**.
- **atol**: une fonction qui convertit une chaîne de caractères en une valeur numérique (valeur de retour) de type **double**.
- **atof**: une fonction qui convertit une chaîne de caractères en une valeur numérique (valeur de retour) de type **float**.
- **Règles générales pour la conversion:**
  - Les espaces au début d'une chaîne sont ignorés
  - Il n'y a pas de contrôle du domaine de la cible
  - La conversion s'arrête au premier caractère non convertible
  - Pour une chaîne non convertible, les fonctions retournent **zéro**

Tableau de chaines de caractères

- Un tableau de chaînes de caractères correspond à un tableau à deux dimensions du type **char**, où ***chaque ligne contient une chaîne de caractères***.
- Déclaration : char JOUR[7][9];
  - réserve l'espace en mémoire pour 7 mots contenant 9 caractères (dont 8 caractères significatifs).
- Lors de la déclaration il est possible d'initialiser toutes les composantes du tableau par des chaînes de caractères constantes:
- char JOUR[7][9]= {"lundi", "mardi", "mercredi", "jeudi", "vendredi", "samedi","dimanche"};

JOUR:

'l'	'u'	'n'	'd'	'i'	'\0'			
'm'	'a'	'r'	'd'	'i'	'\0'			
'm'	'e'	'r'	'c'	'r'	'e'	'd'	'i'	'\0'
...	...	...	...	...	...	...	...	...
'd'	'i'	'm'	'a'	'n'	'c'	'h'	'e'	'\0'

Exercices – part 2

- Ecrire une procédure qui permet de classer par ordre alphabétique une liste de noms fournies comme paramètre à la fonction. Nous devons passez à la procédure la taille de la liste aussi.
- Écrire un programme d'essai. On supposera que chaque mot peut contenir jusqu'à 50 caractères au maximum.