Travaux pratiques n°6:
Chaînes de caractères: Déclaration et utilisation

Objectifs

- Comprendre comment sont représentées les chaînes de caractères en C.
- Être capable de manipuler une chaîne de caractères dans des programmes en C.
- Manipuler les arguments du main.

Partie 1: Compléments pratiques au langage C

Les arguments du main:

Comme il l'a été dit en cours le main est en fait une procédure (la procédure principale d'un programme) et comme d'autres procédures elle possède des arguments. Ainsi main possède comme premier argument un *int* représentant le nombre d'arguments passé en ligne lors de l'exécution et un second qui est un tableau de chaînes représentant les arguments:

Compiler et exécuter le programme *essai.c* suivant en passant des arguments variés:
Par exemple:
./essai 26 étudiants dans ce groupe de TD

```
int main(int nbArg, char* argv[])
{
int i;
printf ("Le nom du programme est %s \n",argv[0]);
if (nbArg != 1)
  for (i=1; i<nbArg; i++)
       printf("\nargument n° %d: %s \n",i,argv[i]);
else printf("Pas d'argument");
}
```

Comme tous les arguments sont sous forme de chaînes il est parfois nécessaire de transformer ces chaînes en d'autres types de base.
Ainsi *atoi(ch);* de la bibliothèque *stdlib* est une fonction qui renvoie l'entier correspondant à la chaîne ch.
exemple:

```
#include<stdlib>
int main(int argc, char* argv[])
{
 int i;
 char ch[10]= "100";
 i = atoi(ch);    /* i est égal à 100 */
 printf("%d", i);  /* affiche  100 */
 return(1);
}
```

Vous pouvez aussi utiliser *atof(ch);* qui convertit la chaîne *ch* en un réel.

Partie 2: Exercices sur les chaînes de caractères en langage C

Remarque: Vous appliquerez le principe de modularité.

Exercice 1:

Vous pouvez utiliser les procédures et fonctions de la bibliothèque *string.h*.

1 Écrire une procédure en C qui insère des tirets bas ('_') entre chaque lettre d'une chaîne. Par exemple, si on lui passe la chaîne "bonjour", le programme retourne la chaîne "b_o_n_j_o_u_r". On suppose le tableau qui contient la chaîne suffisamment grand pour ajouter les tirets.

2 Écrire une procédure en C qui transforme une chaîne en l'inversant. Par exemple, si on lui passe la chaîne "bonjour", le programme retourne la chaîne "ruojnob".

3 Écrire une fonction en C qui indique si une chaîne de caractères représentant une expression parenthésée est syntaxiquement correcte du point de vue des parenthèses ou pas. La fonction renvoie −1 si l'expression est correcte et la position de la première erreur si l'expression est incorrecte.

exemple:

pour "(a.(b))" elle retourne −1 ;
pour "(()())" elle retourne −1 ;
pour "a.(b))(" elle retourne 5 ;
pour "(()(()a.(b))" elle retourne 11 ;
pour ")(" elle retourne 0 ;

Exercice 2: Palindrome

a) Écrire une fonction/procédure qui renvoie vrai si la chaîne passée en argument est un palindrome de phrase (les espaces ne comptent pas) .

exemple:

palindrome("et la marine va venir a malte"); renvoie « vrai ».
palindrome("esope reste ici et se repose"); renvoie « vrai ».

b) Écrire un programme principal (*testPalin.c*) qui affichera « oui c'est un palindrome » (resp. « non ... ») si la chaîne constituée des arguments du programme principal est (resp. n'est pas) un palindrome.

exemple:
./testPalin esope reste ici et se repose
va afficher: oui c'est un palindrome.

Exercice 3: Calcul.

Écrire un programme C (*calculSimple.c*) dont les arguments constituent une expression arithmétique simple (opérande1 opérateur opérande2) et qui affiche le résultat de l'expression. On suppose que tous les arguments sont séparés par un espace.

Par exemple:

./calculSimple 12 + 14

va afficher:

12 + 14 = 26