Algorithmique et programmation procédurale

TP3

Exercice 1 – Chaînes de caractères

- Ecrire un programme poli demandant de saisir un prénom et répondant "Bonjour <prénom saisi>".
- Ecrire une procédure qui remplacer un caractère par un autre dans une chaîne.
- Ecrivez le programme principal demandant la saisie du caractère à remplacer, du caractère de remplacement et le mot à modifier. Testez votre programme, que constatez-vous ?

  Pour vider le buffer en lecture, utilisez le code suivant :

```
do {
     entree = getchar();
} while (entree!='\n');
```

  Que se passe-t-il ? Pourquoi ?
- Ecrivez une fonction qui vérifie qu'une chaîne est bien contenue dans une seconde.
  a) en parcourant les chaînes en tant que tableau
  b) en utilisant la fonction strstr.

Exercice 2 – Carré magique

Un carré magique d'ordre n est un tableau nxn tel que la somme des entiers de chaque ligne, chaque colonne et des deux diagonales est identique. Par exemple

8	1	6
3	5	7
4	9	2

est un carré magique d'ordre 3. Ecrivez une fonction qui teste si un carré est magique et un programme testant votre fonction.

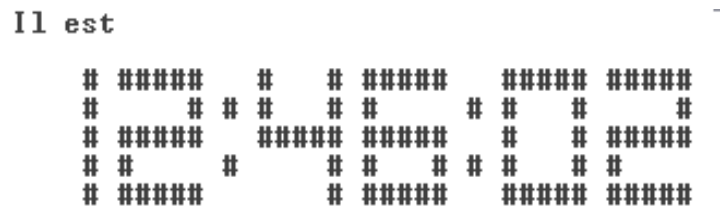
Exercice 3 – Horloge

L'objectif consiste à créer un programme qui affiche l'heure locale en temps réel, jusqu'à ce que le processus soit interrompu (Ctrl-C).

Afin de connaître l'heure locale, servez vous des fonctions de la bibliothèque `time.h` :

```
...
time_t epoch_time = time(NULL); // le nb de sec depuis 1 jan 1970 00h00 UTC
struct tm *tm_p = localtime(&epoch_time); // transformation en heure locale
int h = tm_p->tm_hour; // extraire les heures
int m = tm_p->tm_min;  // extraire les minutes
int s = tm_p->tm_sec;  // extraire les secondes
printf("Il est %.2d:%.2d:%.2d\n", h, m, s);
...
```

Une fois que vous avez réalisé un programme qui affiche l'heure locale de façon continue seconde par seconde, passez à une interface plus sophistiquée :



Utilisez un tableau à deux dimensions pour gérer la matrice des cases « allumées » et « éteintes » de cet affichage.

Notez également : sous Linux, la commande C

```
system("clear"); // system("cls"); sous Windows
```

permet de nettoyer le terminal et de créer ainsi l'illusion d'un affichage stable sur l'écran.