TP N° 3 (une séance)

Exercice 1 :

```c
#include <stdio.h>
void main()
{
   char c;
   for(c=-127; c<127; c=c+1)
   {
      printf("\nAu code ASCII %d correspond le caractère :%c", c, c);
   }
}
```

Que pensez-vous de ce programme?

Exercice 2 :

Ecrire un programme qui convertit une chaîne en majuscule s'il est minuscule et vice-versa. Le programme doit afficher un message d'erreur si un caractère saisi n'est pas une lettre.

Exercice 3 :

Créer un programme permettant d'afficher un triangle isocèle formé d'étoiles de N lignes (N étant fourni au clavier).

Nombre de lignes : 8

```
       *
      *o*
     *ooo*
    *ooooo*
   *ooooooo*
  *oooooooo*
 *ooooooooo*
***************
```

Exercice 4 :

Ecrire un programme qui calcule le produit scalaire de deux vecteurs d'entiers U et V (de même dimension).

Exemple:

```
/          \   /          \
| 3  2  -4 | * | 2  -3  5 | = 3*2+2*(-3)+(-4)*5 = -20
\          /   \          /
```

Exercice 5 :

Un tableau A de dimension N+1 contient N valeurs entières triées par ordre croissant; la (N+1)ième valeur est indéfinie. Insérer une valeur VAL donnée au clavier dans le tableau A de manière à obtenir un tableau de N+1 valeurs triées.

TP N° 4 : Algorithmes de tri

Le problème du tri : On désigne par "tri" l'opération consistant à ordonner un ensemble d'éléments en fonction de clés sur lesquelles est définie une relation d'ordre. Les algorithmes de tri ont une grande importance pratique. Ils sont fondamentaux dans certains domaines, comme l'informatique de gestion où l'on tri de manière quasi-systématique des données avant de les utiliser.

L'étude du tri est également intéressante en elle-même car il s'agit sans doute du domaine de l'algorithmique qui a été le plus étudié et qui a conduit à des résultats remarquables sur la construction d'algorithmes et l'étude de leur complexité.

Exercice 1 : Tri à bulles

Le tri à bulle : L'algorithme parcourt la liste et compare les couples d'éléments successifs. Lorsque deux éléments successifs ne sont pas dans l'ordre croissant, ils sont échangés. Après chaque parcours complet de la liste, l'algorithme recommence l'opération. Lorsqu'aucun échange n'a lieu pendant un parcours, cela signifie que la liste est triée : l'algorithme peut s'arrêter.

Ecrire un programme qui lit un tableau dont la dimension est saisie puis tri ce tableau par ordre croissant. Afficher le tableau après le tri.

Exercice 2 : Tri par sélection

Le tri par sélection (ou tri par extraction) est un des algorithmes de tri les plus triviaux. Il consiste en la recherche soit du plus grand élément (ou le plus petit) que l'on va replacer à sa position finale c'est-à-dire en dernière position (ou en première), puis on recherche le second plus grand élément (ou le second plus petit) que l'on va replacer également à sa position finale c'est-à-dire en avant dernière position (ou en seconde), etc., jusqu'à ce que le tableau soit entièrement trié

Ecrire un programme qui lit un tableau dont la dimension est saisie puis tri ce tableau par ordre croissant. Afficher le tableau après le tri.

Exercice 3 : (Tri par insertion)

Le tri par insertion consiste à parcourir la liste : on prend les éléments dans l'ordre. Ensuite, on les compare avec les éléments précédents jusqu'à trouver la place de l'élément qu'on considère. Il ne reste plus qu'à décaler les éléments du tableau pour insérer l'élément considéré à sa place dans la partie déjà triée.

Ecrire un programme qui lit un tableau dont la dimension est saisie puis tri ce tableau par ordre croissant. Afficher le tableau après le tri.

Exercice 4 : (Recherche Dichotomique)

Ecrire un programme qui affiche un tableau trié dont la dimension est saisie puis cherche une valeur saisi dans le tableau.

N.B : Les instructions suivantes peuvent vous aider la saisie du tableau avec des valeurs aléatoire

```
for(i = 0; i<400; i++)
            printf(" %d ", rand() % (100-5) + 5);
```