Université Hassan 1er
Faculté des Sciences et Techniques
Département Mathématiques et Informatique

# Algorithmique et Programmation en langage C

# Travaux Pratiques

**Professeur BALOUKI Youssef**
**Département Mathématiques et Informatique**

Université Hassan 1er
Faculté des Sciences et Techniques
Département Mathématiques et Informatique

# Travaux Pratiques 1 de la Programmation Avancée en C

**Exercice 1 :**

Ecrire un programme qui lit deux matrices A et B de dimensions N et M respectivement M et P au clavier et qui effectue la multiplication des deux matrices. Le résultat de la multiplication sera affecté à la matrice C, qui sera ensuite affichée. Utiliser le formalisme pointeur à chaque fois que cela est possible.

**Exercice 2 :**

Ecrire un programme qui lit 5 mots d'une longueur maximale de 50 caractères et les mémorise dans un tableau de chaînes de caractères TABCH. Inverser l'ordre des caractères à l'intérieur des 5 mots. Afficher les mots.

**Exercice 3 :**

Ecrire un programme qui lit une phrase de taille maximal 200 caractères au clavier puis la convertir en un tableau de chaine de caractères des mots qui la construisent.

**Exercice 4 :**

Écrire un programme qui construit un tableau FUS trié par ordre ASCII avec les éléments de deux tableaux de chaines de caractères A et B.

**Exercice 5 :**

Ecrire un programme en langage C qui lit un verbe régulier en « er » au clavier et qui en affiche la conjugaison au présent de l'indicatif de ce verbe. Contrôlez s'il s'agit bien d'un verbe en « er » avant de conjuguer. Utiliser les fonctions gets, puts,strcat et strlen.

Université Hassan 1er
Faculté des Sciences et Techniques
Département Mathématiques et Informatique

# Travaux Pratiques 2 de la Programmation Avancée en C

## Exercice 1 :

Écrire une fonction distance ayant comme paramètres 4 doubles xa,ya et xb,yb qui représentent les coordonnées de deux points A et B et qui renvoie la distance AB. Tester cette fonction.

## Exercice 2 :

Déclarer et initialiser une matrice [5,5] d'entiers iMat (l'utilisation des pointeurs est fortement recommandée). Ecrire une fonction afficher_matrice qui admette en paramètre une matrice [5,5] et qui imprime ses éléments sous forme de tableau. La fonction main fera un appel à affiche_matrice pour la matrice iMat.

## Exercice 3 :

Ecrire un programme comportant :

1. La déclaration de trois variables globales entières iHeues, iMinutes, iSecondes.
2. Une fonction affiche_heure qui imprimera la message suivant : Il est … heure(s) … minute(s) … seconde(s) en respectant l'orthographe du singulier et du pluriel.
3. Une fonction saisir_heure qui admettra trois paramètres entiers iH, iM et iS dont elle affectera les valeurs respectivement à iHeures, iMinutes et iSecondes.
4. Une fonction tick qui incrémentera l'heure d'une seconde.
5. La fonction main sera un jeu d'essais des procédures précédentes.

## Exercice 4 : *(Algorithme de cryptage)*

On choisit un décalage (par exemple 5), et un a sera remplacé par un f, un b par g, un c par un h, etc.. On ne cryptera que les lettres majuscules et minuscules sans toucher ni à la ponctuation, ni à la mise en page. On supposera que les codes des lettres se suivent de *a* à *z* et de *A* à *Z*.

1. Déclarer un tableau de caractères *Message* initialisé avec le message en clair.
2. Ecrire une fonction *crypt* de cryptage d'une chaine de caractères.
3. Ecrire une fonction *decrypt* de décryptage de message.
4. Ecrire le programme principal main qui permet de tester les deux fonctions.

Université Hassan 1er
Faculté des Sciences et Techniques
Département Mathématiques et Informatique

# Travaux Pratiques 3 de la Programmation en C

Exercice 1 :

On désire développer un système de consultation des livres d'une bibliothèque universitaire. L'accès à ce système se fait obligatoirement par une authentification. Pour s'authentifier, l'utilisateur doit fournir sont identifiant (CIN, CNE ou son code apogée) et un mot de passe. Dans cet exercice, on utilise les types de données suivant :

```
enum Identifiant { CIN, CNE, APOGEE };
enum Type_Utilisateur { ETUDIANT, ADMINISTRATEUR};
union Nom_Utilisateur { char CIN[8]; char CNE[10]; int APOGEE; };
struct User {
   enum Identifiant id;
   union Nom_Utilisateur login;
   char password[20];
   enum Type_Utilisateur type;
   char nom[20];
   char prenom[20];
   char date_N[10];
   struct user* suivant; };
typedef struct User User;

struct Livre {
   char ISBN[13];
   char titre[50];
   char auteur[50];
   struct Livre* suivant; };
typedef struct Livre Livre;
```

Ecrire les fonctions suivantes :

- d'authentification.
- de déconnexion.
- d'affichage du menu suivant pour un utilisateur de type ETUDIANT :

```
========== Menu ==========
1- Afficher la liste des livres
2- Se deconnecter
Tapez votre choix :
```

- d'affichage du menu suivant pour un utilisateur de type ADMINISTRATEUR :

```
========== Menu ==========
1- Afficher la liste d'utilisateurs
2- Ajouter un utilisateur
3- Afficher la liste des livres
4- Se deconnecter
Tapez votre choix :
```

De plus, écrire toutes les fonctions permettant de fournir les services présentés dans les deux menus.

NB : Pour des raisons de sécurité, on sauvegarde les mots de passe cryptés au lieu des mots de passe originaux. Pour se faire, vous pouvez utiliser les fonctions de cryptage présentés dans l'exercice 4 de la série 2 avec un décalage de votre choix.