INF1005A - PROGRAMMATION PROCÉDURALE

Travail pratique No. 4
Manipulation des fonctions

Objectifs:	Permettre à l'étudiant de comprendre et maitriser le concept des fonctions et de leur utilisation dans le cadre d'un programme Matlab.
Remise du travail:	**Dimanche 24 Mars 2019 avant 23h30.**
Travail préparatoire:	Étude des chapitres 5 et 6. Lecture de l'énoncé.
Documents à remettre:	Les fichiers .m de tous les exercices (via le site moodle).
Directives:	▪ Vous devez utiliser un fichier .m pour chaque exercice. ▪ Les entêtes et les commentaires sont obligatoires dans tous les fichiers .m (1 point). ▪ Affichez toujours un message approprié avant chaque saisie (input) et chaque sortie (output). ▪ Pour effacer toutes les variables en mémoire et nettoyer la fenêtre de commande dans MATLAB, vous pouvez utiliser respectivement les fonctions *clear all* et *clc* au début de chaque fichier. ▪ **Il est interdit d'utiliser le break et continue**.

ATTENTION : Afin d'éviter tout problème lors de la remise du TP, n'attendez pas à la dernière minute pour déposer vos fichiers sur Moodle. Premièrement, déposez souvent des brouillons de votre travail en cours. Deuxièmement, déposez votre version finale au moins **30 minutes** avant l'heure de remise. Également, n'oubliez pas d'appuyer sur le bouton « Soumettre » pour confirmer la soumission de votre travail.

TOLÉRANCE ZÉRO POUR LES RETARDS!

Important : Dans chaque fonction de ce TP, vous devez 1) valider le nombre de paramètres d'entrée et 2) valider les types des paramètres d'entrée (nombre entier, vecteur, chaîne de caractères, etc.). La fonction doit afficher un message d'erreur approprié le cas échéant (en utilisant la fonction error()). À l'exception des messages d'erreur, les fonctions ne doivent faire aucun affichage (sauf pour les fonctions d'affichage demandées). Par ailleurs, vous devez respecter les noms exacts des fonctions demandées. Chaque fonction doit avoir son propre fichier .m portant le même nom que la fonction, tel qu'indiqué dans la convention de Matlab.

Exercice 1 :

Créer une fonction qui prend en paramètre une chaine de caractères et l'affiche encadrée et centrée comme dans l'exemple d'exécution ci-dessous.

```
function [ ] = encadrer(text)
```

Créer un programme principal qui demande à l'utilisateur une chaine de caractères et appelle votre fonction d'affichage.

Par convention, on nomme habituellement le fichier du programme principal `main.m`.

Exemple d'exécution

```
Entrer une chaine de caractère : INF1005A
************
* INF1005A *
************
```

Exercice 2 : Cryptographie visuelle

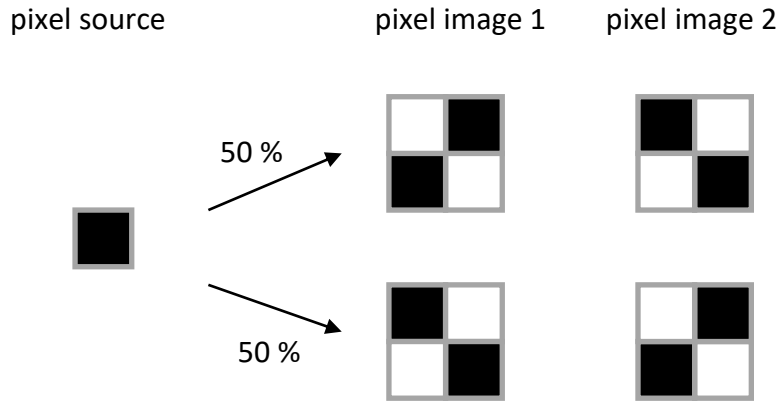
Vous allez implémenter une technique permettant de cacher le contenu d'une image monochrome (en noir et blanc). Le principe est de séparer l'image originale en deux nouvelles images de telle façon que l'on puisse retrouver le contenu original si et seulement si nous superposons exactement les deux images chiffrées. Il est impossible de retrouver le contenu avec une seule image chiffrée.

Dans Matlab, une image de couleur est représentée par trois matrices : une pour la couleur rouge, une pour la couleur verte et une pour la couleur bleue avec, pour chacune, une valeur entre 0 et 255 représentant l'intensité de la couleur pour chaque valeur de la matrice. Un pixel est donc représenté par 3 valeurs, une pour chaque couleur. Dans le cas d'une image monochrome, c'est plus simple. Elle est représentée par une seule matrice, avec comme valeurs pour chaque pixel, 0 ou 1 (0 pour la couleur noire, 1 pour la couleur blanche).

On analyse chaque pixel de l'image originale. Chacun pixel sera converti en un motif de 4 pixels (2 x 2) pour chaque image chiffrée. Les deux images chiffrées résultantes seront donc 4 fois plus grandes que l'originale. De plus, on doit lancer une pièce de monnaie (50 % de chances) afin de déterminer quels motifs créés. Cela est nécessaire afin que les images chiffrées paraissent aléatoires.

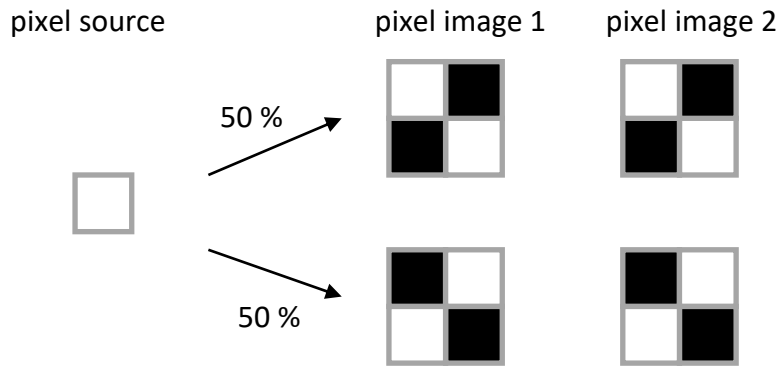
On convertit chaque pixel de la façon suivante :

Pour un pixel noir :



L'important ici est que le motif d'une image chiffrée soit **l'inverse** de celui de l'autre. De cette façon, nous retrouvons le pixel noir original lorsque les deux images chiffrées sont superposées.

Pour un pixel blanc :



On désire ici que le motif créé soit le **même** pour les deux images chiffrées. En superposant les deux images, nous ne retrouvons pas le pixel blanc original. Cependant, ce n'est pas important car le contenu de l'image originale va tout de même être perceptible à l'oeil (voir l'exemple d'exécution).

Vous avez une fonction et un programme principal à coder.

Exercice 2.1 : fonction chiffrer()

Cette fonction crée les deux motifs gauche et droite à partir d'un pixel.

```
function [ motifA motifB ] = chiffrer(pixel)
```

Elle prend en entrée un pixel (0 ou 1) et retourne en sortie deux matrices 2 x 2, contenant elles aussi les valeurs 0 ou 1 correspondant aux motifs créés.

Exercice 2.2 : programme principal

Le programme commence d'abord par demander à l'utilisateur le nom d'une image à charger en mémoire. Utilisez la fonction `imread()` pour charger l'image. Deux images de tests `test.png` et `test2.png` sont fournies. Ensuite, on appelle la fonction `chiffrer()` pour tous les pixels de l'image source. Finalement, on affiche les trois images résultantes : image chiffrée A, image chiffrée B et la combinaison des deux images chiffrées.

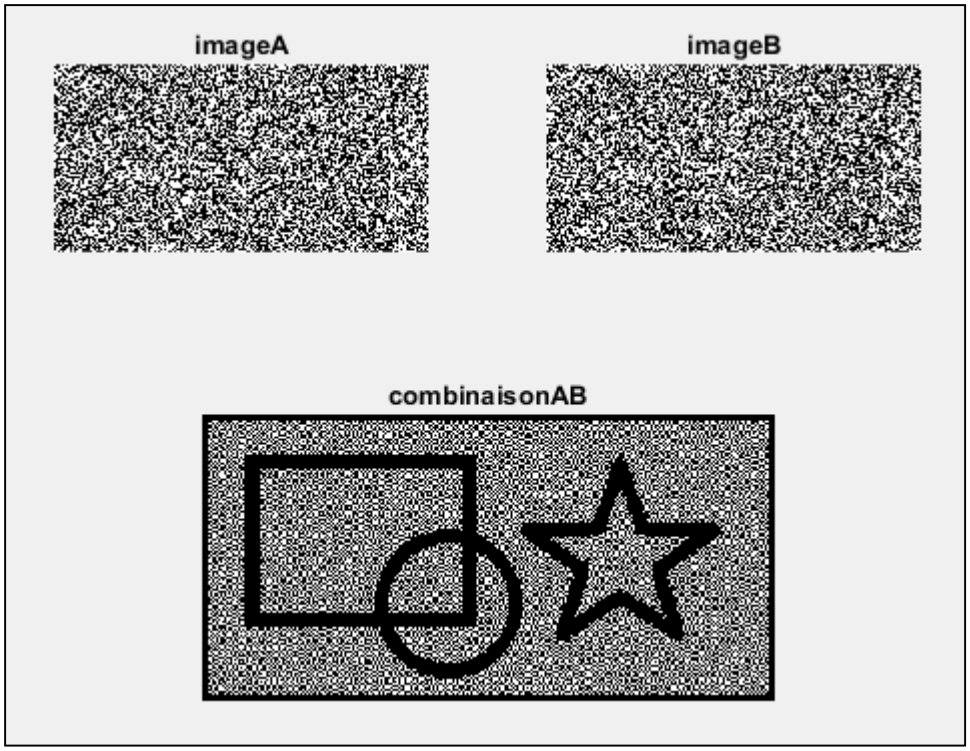
Une fonction d'affichage `affichage.m` vous est fournie. Elle prend en entrées 3 images et les affiche dans une fenêtre comme dans l'exemple d'exécution ci-dessous.

Exemple d'exécution

```
Entrer le nom du fichier image : test.png
```

Image originale :

Résultat :



NOTE : Consultez ce site si vous voulez voir un exemple interactif de la technique.

Exercice 3 : Pointage Scrabble

Exercice 3.1

On vous demande d'implémenter une fonction permettant de calculer le pointage que rapporte un mot dans le jeu de société Scrabble pour diverses langues. Voici le prototype de la fonction :

```
function [pointage] = calculerPointage(mot, langue)
```

où `mot` représente le mot dont on veut calculer le pointage et `langue` représente la langue dans laquelle on calcule le pointage. Vous devez vérifier que chaque paramètre est une chaine de caractères qui ne contient que des lettres (consultez la documentation de la fonction `isstrprop()`) et afficher un message d'erreur si ce n'est pas le cas.

Les grilles de pointage pour l'anglais et le français vous sont fournies (fichiers `francais.mat` et `anglais.mat`). Ces fichiers contiennent une variable `grille` qui est un vecteur de 26 éléments, où chaque valeur correspond au pointage associé à une lettre de l'alphabet. Vous pouvez les charger en mémoire en appelant la fonction Matlab `load()` comme ci-dessous. Une variable `grille` sera automatiquement créée pour vous.

```
load('nomDuFichier.mat', 'grille');
```

La fonction doit charger en mémoire la grille correspondante à la langue spécifiée en paramètre. Si la langue n'est pas spécifiée, le français doit être choisi par défaut. Vous n'avez pas à gérer les accents ou les autres caractères spéciaux, seulement les lettres de l'alphabet.

Exercice 3.2

Maintenant, créer une nouvelle fonction qui prend en paramètre un nombre indéfini de mots en français. Cette fonction fait appel à celle codée en 3.1 . Voici le prototype de la fonction :

```
function [varargout] = calculerPointageTotal(varargin)
```

Créer un programme principal qui appelle vos deux fonctions et affiche le résultat comme dans l'exemple d'exécution ci-dessous.

Exemple d'exécution

```
Le pointage pour le mot communication en francais est : 19
Le pointage pour le mot communication en anglais est : 21

Pointage pour les 3 mots français spécifies :
chaise : 11
table : 7
divan : 9

Pointage pour les 4 mots français spécifies :
banane : 8
poire : 7
pomme : 9
orange : 7
```

Exercice 4 :

NOTE : Pour cet exercice, vous n'avez pas à valider le nombre de paramètres, ni leur type dans vos fonctions.

Vous devez construire un programme qui permet :

- D'authentifier des utilisateurs
- D'enregistrer de nouveaux utilisateurs dans une base de données
- De changer leurs mots de passe
- De supprimer des utilisateurs
- De les déconnecter

Les informations des utilisateurs sont stockées dans une base de données qui a la forme d'un tableau d'enregistrements, dont les champs sont pour une entrée donnée :

- matricule : matricule de la personne en chaine de caractères
- nom : nom de la personne
- prenom : prénom de la personne
- age : âge de la personne
- hashedpw : l'empreinte du mot de passe de la personne en chaine de caractères

Le programme ne stocke jamais les mots de passe des utilisateurs en clair dans la base de données. Il ne stocke que leur empreinte, ou *hash* en anglais. Ainsi, pour vérifier si des mots de passes sont les mêmes, on ne les compare pas directement, on compare les empreintes stockées avec l'empreinte donnée par l'utilisateur.

La fonction calculant les empreintes vous est donnée (`sha1.m`). Cette fonction calcule l'empreinte d'une chaine de caractères en utilisant SHA-1. Elle prend en argument une chaine de caractères et retourne son empreinte sous la forme d'une autre chaine de caractères d'une taille toujours fixe.

Exercice 4.1 : fonction login()

```
function [usager] = login(bd, matricule, mdp)
```

Cette fonction permet de connecter un utilisateur. Elle retourne l'enregistrement correspondant au matricule et au mot de passe spécifié ou un enregistrement vide si l'utilisateur n'a pas été trouvé.

Exercice 4.2 : fonction creerCompte()

```
function [bd, success] = creerCompte(bd, matricule, mdp, nom, prenom, age)
```

Cette fonction enregistre les informations d'un nouvel utilisateur dans la base de données. Elle crée l'enregistrement approprié et l'ajoute à la base de données. On appelle la fonction `sha1()` fournie pour créer l'empreinte du mot de passe fourni en paramètre. Elle retourne une variable `success` indiquant si l'opération a réussie ou non. L'opération échoue si l'utilisateur existe déjà.

Exercice 4.3 : fonction supprimerCompte()

```
function [bd, success] = supprimerCompte(bd, matricule)
```

Cette fonction supprime un utilisateur de la base de données. Elle retourne une variable `success` indiquant si l'opération a réussie ou non. L'opération échoue si l'utilisateur n'a pas été trouvé.

Exercice 4.4 : fonction modifierMdp()

```
function [bd, success] = modifierMdp(bd, matricule, nouveauMdp)
```

Cette fonction permet de modifier le mot de passe de l'utilisateur connecté. Elle retourne une variable `success` indiquant si l'opération a réussie ou non. L'opération échoue si le nouveau mot de passe est le même que l'ancien mot de passe.

Exercice 4.5 : programme principal

Le programme commence par créer la base de données avec un seul utilisateur. Voici ses informations :

Matricule	1337
Nom	sudo
Prénom	root
Age	9000
Mot de passe	pass

Ensuite, le programme principal interagit avec l'utilisateur à l'aide de deux principales interfaces : une pour l'authentification, l'autre pour le menu.

Exemple d'affichage pour l'authentification et le menu

```
Accueil.
1. Authentification
2. Quitter

Faites un choix : 1
Entrez un matricule : 1337
Entrez un mot de passe : pass

Bienvenu root !
1. Afficher mes informations
2. Changer mot de passe
3. Créer un compte
4. Supprimer un compte
5. Déconnexion
6. Quitter

Faites un choix :
```

Utilisez `if/else` et `switch` pour gérer l'affichage de l'interface ainsi que les différentes opérations et les appels aux fonctions. Le programme doit s'exécuter tant que l'utilisateur n'a pas choisi de quitter.

Le programme principal est en charge de demander les différentes informations à l'utilisateur et d'afficher le résultat des opérations. Pour chacune des opérations du menu, il faut afficher un message approprié indiquant si l'opération a réussi ou non. Voici une description du comportement attendu pour chaque opération du menu.

Authentification
On doit demander l'identifiant et le mot de passe à l'utilisateur. Si l'opération réussie, on affiche le menu.

Afficher mes informations
Il faut afficher les informations de l'utilisateur connecté.

Changer mon mot de passe
Il faut demander à l'utilisateur le nouveau mot de passe.

Créer compte
On demande à l'utilisateur toutes les informations relatives à la création d'un compte.

Supprimer compte
On demande à l'utilisateur l'identifiant du compte à supprimer.

Déconnexion
Le programme doit retourner au menu d'authentification.

Quitter
Le programme se termine.

Exemple d'exécution

```
Accueil.
1. Authentification
2. Quitter

Faites un choix : 1
Entrez un matricule : 1337
Entrez un mot de passe : pass

Bienvenu root !

1. Afficher mes informations
2. Changer mot de passe
3. Créer un compte
4. Supprimer un compte
5. Déconnexion
6. Quitter

Faites un choix : 1
Matricule : 1337
Nom : sudo
Prénom : root
Age : 9000
Hash : 9D4E1E23BD5B727046A9E3B4B7DB57BD8D6EE684

1. Afficher mes informations
2. Changer mot de passe
3. Créer un compte
4. Supprimer un compte
5. Déconnexion
6. Quitter

Faites un choix : 2
Entrer le nouveau mot de passe : newpass
Mot de passe modifié.

Faites un choix : 5
Déconnexion.

Accueil.
1. Authentification
2. Quitter

Faites un choix : 1
Entrez un matricule : 1337
Entrez un mot de passe : newpass
```

```
Bienvenu root !

1. Afficher mes informations
2. Changer mot de passe
3. Créer un compte
4. Supprimer un compte
5. Déconnexion
6. Quitter

Faites un choix : 3
Entrer les informations du nouveau compte
Matricule : 123
Mot de passe : 123
Nom : Wong
Prenom : Bill
Age : 99
Compte créé.

1. Afficher mes informations
2. Changer mot de passe
3. Créer un compte
4. Supprimer un compte
5. Déconnexion
6. Quitter

Faites un choix : 5
Déconnexion.

Accueil.
1. Authentification
2. Quitter

Faites un choix : 1
Entrez un matricule : 123
Entrez un mot de passe : 123

Bienvenu Bill !

1. Afficher mes informations
2. Changer mot de passe
3. Créer un compte
4. Supprimer un compte
5. Déconnexion
6. Quitter

Faites un choix : 4
Entrer le matricule du compte à supprimer : 1337
```

```
Compte supprimé.

1. Afficher mes informations
2. Changer mot de passe
3. Créer un compte
4. Supprimer un compte
5. Déconnexion
6. Quitter

Faites un choix : 5
Déconnexion.

Accueil.
1. Authentification
2. Quitter

Faites un choix : 1
Entrez un matricule : 1337
Entrez un mot de passe : newpass
Impossible de se connecter. Recommencez.

Accueil.
1. Authentification
2. Quitter

Faites un choix : 2
Au revoir!
```

Bon travail !