Fonctions interdites

`bin2dec(), break(), cell2mat(), continue(), dec2bin(), exist(), exit(), feof(), max(), numel(), quit(), return(), sort(), strjoin(), sscanf(), strsplit(), struct, sum(), textscan(), textscan(), textread(), true(), unique().`

Aussi interdite `fclose('all')`

Question 1 (3 points)

Choisir tous les réponses qui s'appliquent à chaque énoncé. Vous pouvez avoir plus d'une réponse correcte

1.1 Le nombre $(A7)_{16}$ est

a) $(1010\ 0111)_2$
b) $(128)_{10}$
c) $(247)_8$
d) $(167)_{10}$

1.2 Soit le nombre réel ci-dessous représenté en binaire en simple précision en respectant la norme `IEEE754`

1	1000 0101	0000 0110 0000 0000 0000 000

a) Le nombre dans la base `10` est un nombre négatif
b) Le nombre dans la base `10` est un nombre positif
c) La partie décimale du nombre dans la base `10` est `0.5`
d) Le nombre décimal est `1.5`

1.3 Soient les nombres binaires `int8 :` $(1100\ 0111)_2$ et $(0100\ 0001)_2$. Suite à l'addition de ces deux nombres on obtient :

a) Un débordement
b) Résultat correct de l'addition
c) Une retenue
d) La réponse est un nombre négatif

1.4 Le temps écoulé `temps_ecoule` entre deux moments de temps `t1` et `t2` peut être calculé avec les instructions suivantes :

a)
```
t1 = cputime ;
t2 = cputime ;
temps_ecoule = etime(t2, t1)
```

b)
```
t1 = tic ;
t2 = toc ;
temps_ecoule = t2-t1
```

c)
```
t1 = clock ;
t2 = clock ;
temps_ecoule = etime(t2, t1)
```

d)
```
t1 = tic ;
t2 = toc ;
temps_ecoule = t2
```

1.5 Deux graphiques peuvent être tracées dans la même fenêtre avec les instructions suivantes :

a)
```
plot(x1, y1)
plot(x2, y2)
```

b)
```
plot(x1, y1, x2, y2)
```

c)
```
plot(x1, y1, y2)
```

d)
```
plot(x1, y1)
hold on
plot(x2, y2)
hold off
```

1.6 Le nombre $(256)_{10}$ peut être représenté en MATLAB en utilisant:

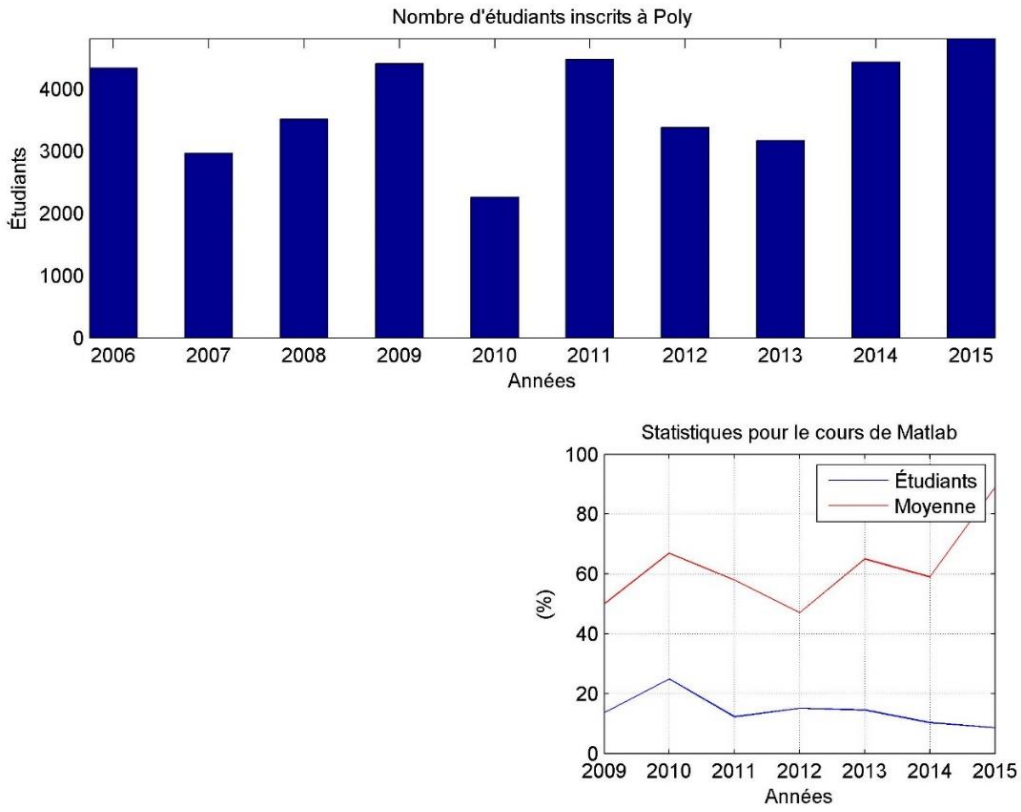
a) int8
b) uint8
c) uint16
d) int16

Reponses
1 – a, c, d
2 – a, c
3 – b, c
4 – c
5 – b, d

6 – c, d

Question 2 (3 points)

Votre ami a été sollicité pour écrire un programme MATLAB afin de reproduire la figure ci-dessous et l'exporter dans le fichier .jpeg `stats`.



1. La première courbe représente le nombre d'étudiants inscrits à Poly, tous cycles confondus, depuis les 10 dernières années
2. La seconde courbe illustre quelques statistiques annuelles pour le cours de MATLAB, depuis l'abolition du tronc commun (2009 à nos jours):
   - Le pourcentage d'étudiants du premier cycle inscrit en MATLAB
   - La moyenne obtenue par ces étudiants.

Avec les données fournies par le registrariat de l'école, votre ami propose le programme suivant, incomplet.

```
1     clc; clear all;
2     % Informations pour les 10 dernières années
3     bac         = [3855 2447 2755 3806 1954 3812 2854 2788 3728 4193];
4     maitrise    = [380 438 529 215 114 354 267 97 368 421];
5     doct        = [100 78 226 380 194 312 254 278 328 193];
6
```

```
7     % Étudiants inscrits en Matlab
8     matlab      = [488 527 538 517 486 467 431 405 385 360];

9     % Moyenne des étudiants inscrits en Matlab
10    moyenne     = [82 87 70 50 67 58 47 65 59 89];
11
12
13    figure(1)
14
15
16
17    bar(annee, sum([bac; maitrise; doct],1), 0.5);
18
19    xlabel('Années')
20    ylabel('Étudiants')
21    title('Nombre d''étudiants inscrits à Poly')
22    axis('tight')
23
24    subplot(2,2,4);
25
26
27
28
29    plot(annee,moyenne, 'r-');
30    xlabel('Années')
31    ylabel('(%)')
32    title('Statistiques pour le cours de Matlab')
33
34
35
36
37
```

À cours de temps, il vous demande de **compléter** le programme, en ajoutant les instructions manquantes. Il vous demande d'indiquer, dans le tableau ci-dessous, le numéro de ligne et l'instruction à ajouter à cette ligne. De plus, il vous indique que l'axe des y sur la seconde figure doit être entre 0 et 100.

Ligne	Instructions

RÉPONSE

Ligne	Instructions
11	`annee = 2006:2015;`
15	`subplot(2,2,1:2)`
26	`plot(annee, 100*matlab./bac);`
27	`hold on;`
33	`legend('Étudiants', 'Moyenne');`
34	`axis([2009 2015 0 100]);`
35	`grid on;`
37	`print -djpeg -r300 stats;`
36	`hold off`

Question 3 (8 points)

3.1 Soit le fichier binaire `nombres.bin` qui contient des nombres binaires, sur 8 bits en complément à deux.

Écrivez un programme MATLAB qui :

- lit le fichier `nombres.bin`
- fait l'addition en binaire des nombres (le premier nombre avec le deuxième, le troisième nombre avec le quatrième, le cinquième avec le sixième, etc) et
- sauvegarde dans le fichier texte `resultats.txt`, sur chaque ligne, les deux nombres binaires, le résultat de l'addition (sur 8 bits) et
    - s'il y a une retenue `R=1` ou sinon `R=0`
    - et/ou s'il y a un débordement `D=1` ou sinon `R=0` et
    - si la réponse de l'addition est correcte ou non.

Notes :

Vous devez vous assurer que :

- le fichier `resultats.txt` n'est pas effacé à chaque fois qu'il est ouvert pour y ajouter de l'information.
- les fichiers soient ouverts, sinon le programme donne une erreur et l'exécution s'arrête.

Vous devez utiliser une structure de répétition et une structure de décision imbriquée dans la structure de répétition pour faire les additions.

Les nombres binaires se trouvant dans le fichier `nombres.bin` ont une taille de 8 caractères de valeur 0 ou 1 (pas de validation requise).

On suppose que le nombre de nombres binaires écrits dans le fichier binaire est pair (pas besoin de le vérifier).

Exemple de fichier `resultats.txt`

```
10010111    00110101    11001100    R=0 D=0 reponse correcte
11001010    10001010    01010100    R=1 D=1 reponse incorrecte
01010101    01000010    10010111    R=0 D=1 reponse incorrecte
11000001    11000010    10000011    R=1 D=0 reponse correcte
11000000    01000100    00000100    R=1 D=0 reponse correcte
```

```
Solution 3.1
clear all; clc
fic=fopen('nombres.bin','r');
fic2=fopen('resultats.txt','at');
if fic~=-1 || fic2~=-1
nb1=fread(fic,[1 8],'*char') ;
  while ~isempty(nom1)
     nb2= fread(fic,[1 8],'*char')
     cin=0;
     for i=8:-1:1
        c(i)=str2num(nb1(i)) + str2num(nb2(i)) + cin
        if c(i)==3 %%ou switch c(i)
                   %%case 3
            cin=1;
            c(i)='1';
        elseif c((i))==2 %%case 2
            cin=1;
            c(i)='0';
        elseif c((i))==1 %%case 1
            cin=0;
            c(i)='1';
        else             %% otherwise ou case 0
            cin=0;
            c(i)='0';
        end
     end
     if (nb1(1)==nb2(1)) && (nb1(1)~=c(1))
         if cin==1
           fprintf(fic2,'%s\t%s\t%s\tR=1\tD=1\trep inc\n', nb1,nb2, c
         else
           fprintf(fic2,'%s\t%s\t%s\tR=0\tD=1\trep inc\n', nb1,nb2, c)
         end
      else
         if cin==1
```

```
            fprintf(fic2,'%s\t%s\t%s\tR=1\tD=0\trep cor\n', nb1,nb2, c)
          else
            fprintf(fic2,'%s\t%s\t%s\tR=0\tD=0\trep cor\n', nb1,nb2, c)
          end
        end
        nom1=fread(fic,[1 8],'*char');
      end
      fclose(fic);
      fclose(fic2);
else
      error('problemes lors de l''ouverture')
end
```

3.2 Statistique Météo vous demande d'écrire un programme MATLAB qui manipule des informations sur les températures moyennes autour du monde. Pour le faire vous avez disponibles deux fichiers texte : `temperatures.txt` et `stations.txt`.

Le fichier texte `temperatures.txt` contient sur chaque ligne 14 nombres réels. Les deux premiers nombres sont les coordonnées d'une station météorologique (la latitude et la longitude) et les 12 nombres suivants représentent les températures moyennes mensuelles mesurées à cette station dans les dernières 30 années. Les mois sont considérés en ordre croissant de janvier à décembre.

Exemple de fichier texte `temperatures.txt`:

```
45.5 -73.7 -10.2 -8.4 -2.3 5.7 13.4 14.2 20.2 21.6 12.6 8.1 1.6 -6.3
49.2 -123.2 3.3  4.8  6.6  9.2 12.5 15.2 20.2 20.6 14.6 10.1 6 3.5
       . . . . . .
37.37 -122.23 5 7   8   8   10  11  12  13  13  11  8   7
```

Le fichier texte `stations.txt` contient sur chaque ligne le nom de stations météo où les températures ont été mesurées (et écrites dans le fichier texte `temperatures.txt`). Sur chaque ligne, on trouve

- le nom de la station météo (une seule chaîne de caractères),
- le pays où la station est située météo (une seule chaîne de caractères),
- la latitude et la longitude de cette station météo. La latitude et la longitude sont des nombres réels et sont toujours les deux derniers éléments écrits sur une ligne.

Exemple de fichier texte `stations.txt`:

```
Vancouver Canada 49.2 -123.2
      . . . . . .
San_Francisco US 37.37 -122.22
     . . . . . .
Montreal Canada 45.5 -73.7
     . . . . . .
```

Toutes les stations qui ont les coordonnées dans le fichier texte `temperatures.txt` se trouvent dans le fichier texte `stations.txt` mais non nécessairement dans le même ordre.

Ecrivez un programme qui sauvegarde dans le fichier binaire `statistiques.bin` seulement les noms et les pays des stations qui se trouvent dans le fichier texte `stations.txt` ainsi que, pour chaque station, les températures moyennes mensuelles mesurées à cette station dans les derniers 30 années (écrites dans le fichier texte `temperatures.txt`).

Le fichier binaire `statistiques.bin` doit pouvoir être lu.

Solution

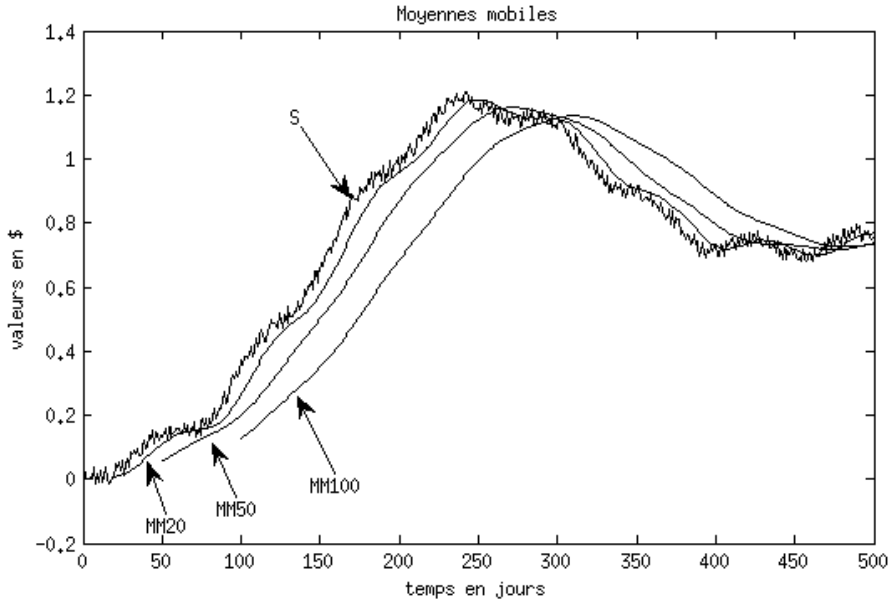
```
idl1=fopen('temperatures.txt', 'rt');
idl2=fopen('stations.txt', 'rt');
idw = fopen('stats.bin', 'w');
if idl1~=-1
    matrtemp= fgetl(idl1);
    j=1;
    while ischar(matrtemp)
        aa=str2num(matrtemp);
        for i=1:14
            matr(j, i) = aa(i);
        end
        j=j+1;
        matrtemp = fgetl(idl1);
    end
    fclose(idl1)
else
    error('erreur')
end
if idw ~=-1 && idl2 ~= -1
    ville=fscanf(idl2, '%s', 1);
    while ~isempty(ville)
        pays=fscanf(idl2, '%s', 1);
        lat=fscanf(idl2, '%f', 1);
        long=fscanf(idl2, '%f', 1);
        for ii=1:size(matr, 1)
            if isequal(lat, matr(ii, 1)) && isequal(long, matr(ii, 2))
                fwrite(idw, length(ville), 'uint8')
                fwrite(idw, ville, 'char')
                fwrite(idw, length(pays), 'uint8')
                fwrite(idw, pays, 'char')
                for i=1:12
                    fwrite(idw, matr(ii, i), 'float32')
                end
            end
        end
        ville = fscanf(idl2, '%s', 1);
    end
    fclose(idl1)
    fclose(idw)
```

```
else
    error('erreur')
end
```

Question 4 (6 points)

Cette question porte sur le concept des moyennes mobiles. Ce concept apparaît souvent en traitement de signal, par exemple sur des signaux sismiques ou financiers. Ces moyennes sont des indicateurs de tendance sur un horizon de temps donné, en se basant sur un horizon de temps passé.

Exemple sur un signal financier (cours de bourse) :



Sur le graphique ci-dessus, la courbe S bruitée représente un cours de la bourse, mesuré quotidiennement (1 valeur par jour). Les trois courbes continues (MM20, MM50 et MM100) sur ce graphique sont les moyennes mobiles à un horizon K de temps passé : K=20 jours, K=50 jours et K=100 jours.

Pour MM20, la valeur de cette courbe à un jour j, est égale à la moyenne des valeurs des cours de bourses sur les 20 jours précédents (jour j inclut, donc la moyenne sur les valeurs des jours j à j-19). Autrement dit pour la MM20 :

$$MM20(j) = \frac{1}{20} \sum_{t=j}^{j-19} S(t)$$

C'est pourquoi sur la figure ci-dessus, les trois moyennes mobiles ne commencent pas au jour j=0, mais aux jours j=20, j-50 et j=100 respectivement.

4.1 Écrivez une fonction qui calcule les moyennes mobiles. Le prototype de la fonction est :

```
function [varargout] = mm(S, varargin)
```

où

- S est le signal d'entrée, donc un vecteur de valeurs réelles ;
- `varargin` contient une série de N valeurs entières représentant les différentes valeurs d'horizons (par exemple, dans l'exemple de la figure, `N=3 et varargin={20, 50, 100}`);
- `varargout` contient les valeurs des N courbes, donc une série de N vecteurs, chacun contient les valeurs des courbes des moyennes mobiles, dans l'ordre chronologique;
- les calculs de moyenne sont faits en utilisant la fonction MATLAB `mean()` ;
- votre fonction doit afficher un message d'erreur approprié :
  - si l'utilisateur ne fournit aucune valeur d'horizon de temps passé;
  - si le signal d'entrée S a moins d'éléments que le plus petit des horizons donnés (car cette situation conduirait au fait qu'aucune moyenne mobile ne pourrait être calculée) ;
  - si les valeurs d'horizons fournies ne sont pas toutes des entiers ;
- votre fonction doit afficher un message d'avertissement :
  - si une valeur d'horizon est supérieure au nombre de valeurs dans le signal S, un message doit indiquer que cette moyenne mobile ne pourra pas être calculée. Le vecteur pour cette moyenne mobile dans `varargout` devra être un vecteur vide.

<u>Remarque importante :</u>
Chaque valeur d'une moyenne mobile est calculée sur les K jours précédents d'un signal. Donc la valeur de cette moyenne mobile ne peut pas être calculée pour les K-1 premières valeurs du signal. Donc les tailles de vos vecteurs de sortie contenus dans `varargout` dépendent des valeurs contenues dans `varargin`. Autrement dit, les valeurs des moyennes mobiles commencent au $K^{ième}$ jour du signal S.

Solution

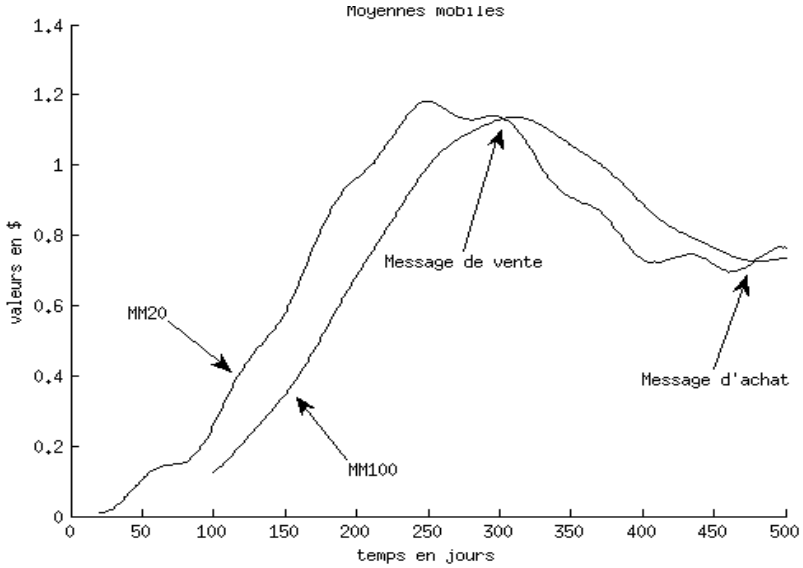
```
function [varargout] = mm(S,varargin) %donnee

for i = 1:length(varargin)%pas obligatoire,
    horizons(i) = varargin{i}
end
if min(size(S)) > 1
    error('Le signal doit être un vecteur');
elseif nargin < 2
    error('Pas assez de paramètres d''entrée');
elseif length(S) < min(horizons)
    error('Les horizons sont tous plus grands que la taille du signal');
elseif any(horizons - floor(horizons))
    error('Certain des horizons ne sont pas des entiers');
end
for i=1:length(horizons)
    K = horizons(i);
    if K > length(S)
        warning('La moyenne mobile d''horizon %i ne peut pas être
calculée car le signal est de longueur inférieure',K);
        varargout{i} = [];
    else
        vMM = zeros(length(S)-K+1,1);
        for j = K:length(S)
            vS = S(j-K+1:j);
            vMM(j-K+1) = mean(vS);
        end
        varargout{i} = vMM;
    end
end
```

4.2 Les moyennes mobiles servent à analyser les cours de la bourse, et savoir quand il est stratégique d'acheter ou vendre les actions représentées par le signal S. Voici une stratégie classique souvent utilisée:

- Si une moyenne mobile court terme franchit à la hausse une moyenne mobile long terme, cela donne un massage d’achat.

- Si une moyenne mobile court terme franchit à la baisse une moyenne mobile long terme, cela donne un message de vente.

Cette stratégie est illustrée dans la figure ci-dessous:



Écrivez une fonction `analyse.m` mettant en place cette stratégie. Le prototype de la fonction est :

`function [] = analyse(mmC,mmL)` où

- `mmC` et `mmL` sont deux vecteurs de moyennes mobiles. `mmC` est la moyenne mobile à court terme et `mmL` la moyenne mobile à long terme. Vous supposerez dans cette question qu'ils sont de même longueur et qu'ils démarrent le même jour (normalement, il faudrait écrire un programme préliminaire pour s'occuper de cette étape, mais il n'est pas demandé dans l'examen). C'est à dire que mmC(j) et mmL(j) représentent les valeurs de ces courbes au même jour j.
- chaque signal d'achat ou de vente détecté sur ces vecteurs doit générer un affichage indiquant le type de signal (achat ou vente) et la valeur du temps où ce signal se produit.

Solution 4.2

```
function []=analyse(mmC,mmL) % donnee
for i=2:length(mmC)
  if mmC(i) > mmC(i-1) && mmC(i) >= mmL(i) && mmC(i-1) <= mmL(i-1)
      fprintf('Signal d''achat au temps t=%i\n',i);
  elseif mmC(i) < mmC(i-1) && mmC(i) <= mmL(i) && mmC(i-1) >= mmL(i-1)
        fprintf('Signal de vente au temps t=%i\n',i);
    end
end
```