Nom/prénom : no camipro :

ME-213, Programmation pour ingénieur
Test Final 2022 + Corr.

Durée 1h45min.

Merci de signer cette page.

Aucun document, ni appareil électronique (calculette, smartphone, etc.) n'est autorisé. Vous pouvez utiliser un crayon et/ou un stylo et répondre en français ou anglais. Vos réponses doivent être lisibles !

Aucune feuille volante n'est reprise, seules les feuilles distribuées seront reprises. Vous pouvez utiliser le verso des pages comme brouillon.

Si vous devez faire des hypothèses, notez-les.

No place

Nom

Camipro

Signature :

Nom/prénom : no camipro :

1. (20 pts) La matrice carrée **M**[max][max] contient des '0' et des '1'.
Ex :

```
int M[max][max]= {
     {0,0,0,0,0,0,0},
     {0,0,0,0,0,0,0},
     {0,0,1,0,0,1,0},
     {0,0,1,0,0,0,0},
     {0,0,0,0,0,0,0},
     {0,0,0,0,0,0,0},
     {0,0,0,0,0,0,0}
};
```

La fonction **isTriangle(M)** détermine s'il existe trois '1' formant un triangle et indique de quel type de triangle il s'agit.

- S'il y a plus, ou moins, de trois '1', **isTriangle(M)** retourne -1 ;

- Si les trois '1' sont alignés (vertical, horizontale, oblique), i.e. pentes identiques, **isTriangle(M)** retourne 0;

```
...
{0,0,1,0,0,0,0},
{0,0,0,0,1,0,0},
{0,0,0,0,0,0,1},
...
```

- Si les trois '1' forme un triangle rectangle dont l'un des bords est horizontal (4 possibilités), **isTriangle(M)** retourne 1;

```
...
{0,0,1,0,0,0,0},    {0,1,0,0,0,1,0},    {0,0,0,0,0,1,0},    {0,1,0,0,0,1,0},
{0,0,0,0,0,0,0}, ou {0,0,0,0,0,1,0}, ou {0,0,0,0,0,0,0}, ou {0,1,0,0,0,0,0},
{0,0,1,0,0,0,1},    {0,0,0,0,0,0,0},    {0,1,0,0,0,1,0},    {0,0,0,0,0,0,0},

...
```

- Si les trois '1' forme un triangle quelconque, **isTriangle(M)** retourne 2;

```
...
{0,0,1,0,0,0,0},
{1,0,0,0,0,0,0},
{0,0,0,0,0,0,1},
...
```

Ecrire en C la fonction :

```
int isTriangle (int M[max][max]);
```

qui retourne le type de triangle trouvé ou une erreur s'il n'y a pas exactement trois '1'.

La constante **max** est définie en dehors de votre fonction, ex. #define max 7

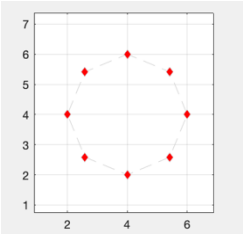
Nom/prénom : no camipro :

```
int isTriangle(int M[max][max]){
    int X[3], Y[3];
    int cnt =0;

    for (int L=0; (L<max);L++)
        for (int C=0; (C<max) ; C++)
            if (M[L][C] == 1) {
                if (cnt<3) {
                    Y[cnt]=L;
                    X[cnt]=C;
                }
                cnt++;
            }

    if (cnt!=3) return -1; // too many/few '1'

    if ( (double)(Y[0]-Y[1]) / (X[0]-X[1]) == (Y[0]-Y[2]) / (double)(X[0]-X[2]))   return 0; // 3pts aligned

    if (     (  (X[0]==X[1]) || (X[1]==X[2]) || (X[2]==X[0]) )              // 1 vert + 1 horiz -> rect
        &&   (  (Y[0]==Y[1]) || (Y[1]==Y[2]) ||( Y[2]==Y[0]) )  )   return 1;// 2 x v/h handled before

    return 2; // default case
}
```

Nom/prénom : no camipro :

---

2. (6 pts) Ecrire en matlab, **sans boucle,** la fonction **BuildOcto**(cote, centre) qui retourne les vecteurs **x** et **y** contenant les coordonnées des 8 points formant un octogone comme sur la figure ci-contre :



function [x y]= **BuildOcto**(cote, centre)

avec

cote : la taille d'un côté de octogone,
centre *(vecteur de 2 val.)* : les coordonnées du centre de octogone.

```
function [x y]= BuildOcto(cote, center)
      rayon = cote/(2* sin(pi/8));
      angles = 0:pi/4:14*pi/8;
      % or angles = linspace(pi/4,2*pi,8);
      y = rayon * cos(angles) + center(1);
      x = rayon * sin(angles) + center(2);
end
```

---

3. (4 pts) Ecrire en matlab, **sans boucle**, la fonction anonyme **Serie1** (*n*) qui retourne les *n* premières valeurs de le série suivante : -1 2 -3 4 -5 6 -7 …

Ex. **Serie1**(5) retourne [-1 2 -3 4 -5]

```
Serie1= @(n) (1:n).*(-1).^(1:n)
Serie1= @(n) (1:n)-2.*(1:n).*mod([1:n],2)
Serie1= @(n) (1:n).*(mod(1:n,2) -0.5) *2
Serie1= @(n) (1:n).*cos((1:n)*pi)
```

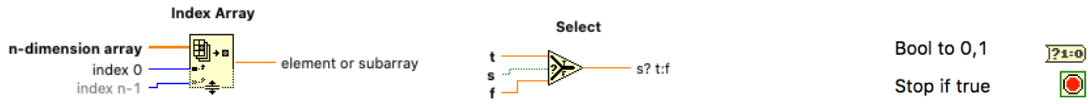
---

4. (4 pts) Ecrire en matlab, **sans boucle**, la fonction anonyme **Serie2** (*n*) qui retourne les *n* premières valeurs de le série suivante : 0 4 0 16 0 36 0 64 …

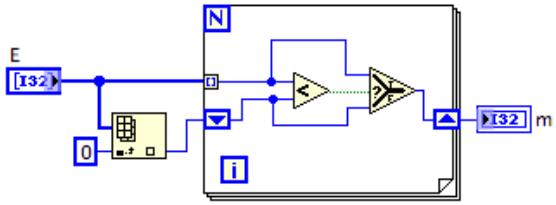
Ex. **Serie2**(4) retourne [0 4 0 16]

```
Serie2= @(n)  (mod(1:n,2)).*((1:n).^2)
Serie2= @(n)  (1:n).*(1:n).*mod([1:n]+1,2)
```

Nom/prénom : no camipro :



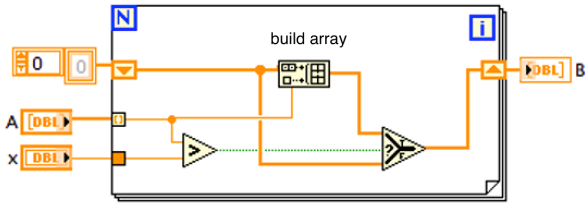
5. (2 pts) Ecrire en matlab (1 ligne, sans boucle for) le code équivalent au *diagram* ci-dessous pour **m** fonction de **E**



m=min(E)

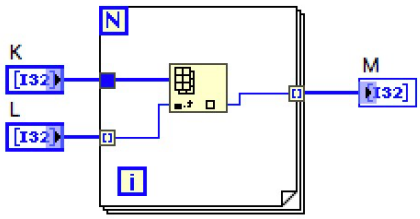
---

6. (2 pts) Ecrire en **Matlab** (1 ligne, sans boucle for) le code équivalent au *diagram* ci-dessous



B=A(find(A>x))

7a. (2 pts) Si **K** = [7 5 3 1] et **L** = [2 0 1 2], que vaut **M** après l'exécution du *diagram* ci-dessous.



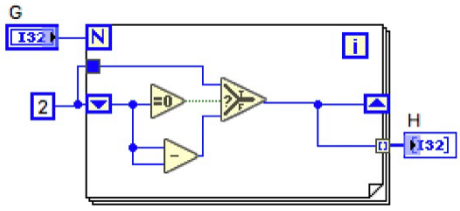
M = [3 7 5 7]

7b. (2 pts) Ecrire en matlab (1 ligne, sans boucle for)) le code équivalent au *diagram* ci-dessus pour **M** fonction de **K** et **L**

M=K(L+1)

Nom/prénom : no camipro :

8a. (2 pts) Si **G** vaut 4, que vaut **H** après l'exécution du *diagram* ci-dessous..



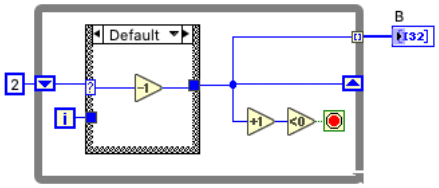
H = [0 2 0 2]

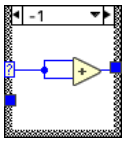
8b. (2 pts) Ecrire en matlab (1 ligne, sans boucle for)) le code équivalent au *diagram* ci-dessus pour **H** fonction de **G**

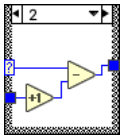
H = 2* mod([1 :G]-1,2)
H = 1+(-1).^(1:G)
H (2 :2 :G) = 2

---

9. (6 pts) Pour la machine d'état ci-après, que vaut **B** après l'exécution du *diagram* ci-dessous. Montrez vos étapes intermédiaires.







autres frames du switch

B = [1 0 -1 -2]