Exercices pour l'utilisation de pointeurs

Exercice 1 (à moitié corrigé)

Cet exercice permet d'apprehender les fonctions et le passage par pointeurs. Lorsqu'on créée une matrice MxM, on peut l'envisager de deux façons :

- Soit comme un simple pointeur avec des allocations `float *Mat = new float [M*M]`
- Soit comme un double pointeur `float **Mat = new float*[M]; for(int i=0;i<M;i++) ...`

Pour chaque fonction, demandez vous quelles sont les entrées et sorties. Sachez qu'il existe plusieurs solutions.
Faire un main et une fonction qui permette de:

- créer une fonction `eye` qui prend en paramètre une matrice MxM, et qui la remplit telle que cette matrice devienne la matrice identité
- créer une fonction `somme` qui prend en paramètre deux matrices MxM, et qui en fait la somme.
- Une fonction `Affiche` qui prend en paramètre une matrice MxM et qui l'affiche en allant bien à la ligne à chaque fin de ligne.
- Une fonction `minmax` qui renvoie la valeur max et min d'une matrice.
- Un main qui utilise ces fonctions

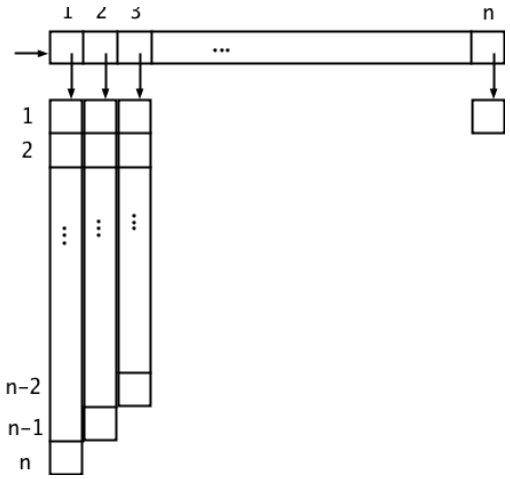
La solution est donnée ci-dessous pour les pointeurs simples. A vous d'essayer de les implanter avec des pointeurs doubles.

Solution de l'exercice pour les pointeurs simples

```
void Eye(float* Mat, int M)
{ for (int i=0; i<M;i++)
{for (int j=0; j<M;j++)
{
    if (i==j)
    {*(Mat+i*M+j)=1;}
    else
    {*(Mat+i*M+j)=0;}
}
}
}



void somme(float* Mat1, float* Mat2, float* Res, int M)
{ for (int i=0; i<(M*M);i++)
    {* (Res + i)= * (Mat2 + i)+ * (Mat1 + i);
    }
}

float* somme2(float* Mat1, float* Mat2, int M)
{
    float *Res=new float[M*M];
    for (int i=0; i<(M*M);i++)
{* (Res + i)= * (Mat2 + i)+ * (Mat1 + i);
}
    return Res;
}



void Affiche(float* Mat, int M)
{ for (int i=0; i<M;i++)
    {for (int j=0; j<M;j++)
    {
        cout<< *(Mat+i*M+j);
}
        cout<<endl;
}
}

int main(){

    int M;
    cout<<"M?";
    cin>> M;

    float * Mat1=new float[M*M];
    float * Mat2=new float[M*M];
    float * MatRes=new float[M*M];
    float * MatRes2;



    Eye(Mat1,M);
    Eye(Mat2,M);
    somme(Mat1,Mat2,MatRes,M);

    MatRes2=somme2(Mat1,MatRes,M);

    Affiche(MatRes,M);
    Affiche(MatRes2,M);

    delete [] Mat1;
    delete [] Mat2;
    delete [] MatRes;
    delete [] MatRes2;
    return(0);

}
```

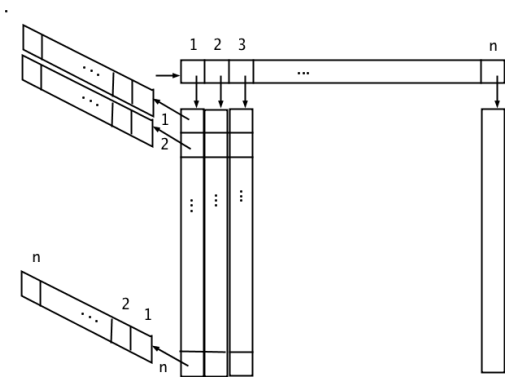
Exercice 2 (Non corrigé)

Créez, grace à des doubles pointeurs, une structure triangulaire comme ci-dessous. Remplissez-la, calculez la somme de tous les nombres. Désallouez la.



Exercice 3 (corrigé)

Créez, grace à des triples pointeurs, une structure cubique comme ci-dessous. Remplissez-la, calculez la somme de tous les nombres. Désallouez la.



Solution

```
int *** tab;
int n,i,j,k;
cout<<"entrez la taille"<<endl;
cin>>n;

//allocation complete
tab=new int**[n];
int somme=0;

for(i=0;i<n;i++)
{
    *(tab+i) =new int*[n];
}

 for(i=0;i<n;i++)
 {
      for(j=0;j<n;j++)
      {
          *(*(tab+i)+j)=new int[n];
      }
 }

//remplit le cube
for(i=0;i<n;i++)
    for(j=0;j<n;j++)
        for(k=0;k<n;k++)
            *(*(*(tab+i)+j)+k) = i+j+k;

//calcule la somme du cube
for(i=0;i<n;i++)
    for(j=0;j<n;j++)
        for(k=0;k<n;k++)
            somme+=  tab[i][j][k]; // equivalent de somme=somme+ tab[i][j][k];


cout<<"Somme="<<somme<<endl;

//desallocation
for(i=0;i<n;i++)
{
    for(j=0;j<n;j++)
    {
       delete []  *(*(tab+i)+j);
    }
}


for(i=0;i<n;i++)
{
    delete [] *(tab+i);
}

delete [] tab;
```

Exercice 4 :Exercice colle 2022 sur 5 points (corrigé)

Soit, à disposition, les 6 fonctions suivantes :

```
void Echange1(int a,int b)
{
int dummy;
dummy=a;
a=b;
b=dummy;
}
void Echange2(int & a, int & b)
{
int dummy;
dummy=a;
a=b;
b=dummy;
}
void Echange3(int & a, int & b)
{
cout« "je ne fais rien" «endl;
}
```

```
void Echange1(int* pa,int* pb)
{
int* dummy;
dummy=pa;
pa=pb;
pb=dummy;
}
void Echange2(int * & pa, int * & pb)
{
int* dummy;
dummy=pa;
pa=pb;
pb=dummy;
}
void Echange3(int * pa, int * pb)
{
int dummy;
dummy=*pa;
*pa=*pb;
*pb=dummy;
}
```

QUESTION : Qu'affichera ce main, qui affiche a, b , *pa et *pb après chaque échange

```
int main() {
int a= 3;
int b= 1;
int *pa= &a;
int *pb= &b;
Echange1(a,b); cout«a«","«b«","«*pa«","«*pb«endl;
Echange1(pa,pb); cout«a«","«b«","«*pa«","«*pb«endl;

Echange2(a,b); cout«a«","«b«","«*pa«","«*pb«endl;
Echange2(pa,pb); cout«a«","«b«","«*pa«","«*pb«endl;

Echange3(&a,&b); cout«a«","«b«","«*pa«","«*pb«endl;
Echange3(pa,pb); cout«a«","«b«","«*pa«","«*pb«endl;
return 0;
}
```

Solution

Echange1(a,b); cout«a«","«b«","«*pa«","«*pb«endl;// 3,1,3,1 (il n'y a pas d'echange, car pas de passage par reference ni par pointeur)

Echange1(pa,pb); cout«a«","«b«","«*pa«","«*pb«endl;// 3,1,3,1 (il n'y a pas d'echange, ce sont des copies de pa et de pb qui sont echangées, pas les pointeurs pa et pb)

Echange2(a,b); cout«a«","«b«","«*pa«","«*pb«endl; 1,3,1,3 (les variables a et b sont echangées (passage par référence))

Echange2(pa,pb); cout«a«","«b«","«*pa«","«*pb«endl; 1,3,3,1 (les pointeurs pa et pb sont echangés, car ils sont passés par référence)

Echange3(&a,&b); cout«a«","«b«","«*pa«","«*pb«endl; 3,1,1,3 (&a et &b, sont les adresses de a et b. On utilise donc un passage par pointeur qui echange les

variables a et b. Ici ce n'est pas la fonction void Echange3(int & a, int & b) qui est utilisée, mais void Echange3(int * pa, int * pb))

Echange3(pa,pb); cout«a«","«b«","«*pa«","«*pb«endl; 3,1,1,3 (a et b, sont echangées grâce à un passage par pointeur)