LIF 1 – Jeudi 8 novembre 2012
Durée 1h

Les documents, calculatrice, téléphone portable et autres appareils de communication sont interdits
La qualité de l'écriture et de la présentation sera prise en compte dans la note finale.
Vous veillerez à respecter les notations et les règles d'écriture des algorithmes vues en cours et en TD.
Le barème est donné à titre indicatif.

Partie A : Questions de cours (4 points)

Soit le programme C/C++ suivant :

```
#include <iostream>
using namespace std;
const int MAX= 7;

void mystere (char c1, char c2)
{ int i,j;
    for (i=0;i<MAX;i++)
    {  for (j=0;j<MAX;j++)
         if ((i==j) || (i==MAX-j-1)) cout<<c1;
            else cout<<c2;
        cout<<endl;    }
}
int main (void)
{    mystere('x','o');
    return 0;
}
```

Donnez le résultat de l'affichage produit lors de l'exécution de ce programme.

Partie B : Algorithmique (8 points)

La suite de Lucas est définie de la manière suivante :

$$\mathcal{L}_0 = 2$$
$$\mathcal{L}_1 = 1$$
$$\mathcal{L}_{n+2} = \mathcal{L}_{n+1} + \mathcal{L}_n$$

1- Écrire en langage algorithmique une **fonction** qui calcule et retourne le nième terme de la suite de Lucas, **sans utiliser la récursivité, ni mémoriser les termes précédents dans un tableau**.
2- Écrire en langage algorithmique une **fonction** qui effectue la saisie d'un nombre entier **appartenant** à l'intervalle [a, b] avec a et b des paramètres de la fonction et ne renvoie la valeur que lorsqu'elle est bien dans l'intervalle.
3- Écrire en langage algorithmique le programme principal qui remplit un tableau de taille maximale 12 avec les n premiers termes de la suite de Lucas. On vérifiera au préalable que la valeur de n saisie par l'utilisateur est bien dans l'intervalle [3,12]. On réutilisera les fonctions précédentes.

Partie C : Langage C (8 points)

1- Écrire en langage C/C++ l'instruction permettant de déclarer une constante MAX valant 5
2- Écrire en langage C/C++ une procédure `remplir_tab` permettant de remplir un tableau 2D de taille MAX*MAX avec des valeurs données par l'utilisateur
3- Écrire en langage C/C++ une fonction booléenne `sans_zero` permettant de vérifier qu'un tableau de taille MAX*MAX d'entiers passé en paramètre ne contient aucun 0
4- Écrire en langage C/C++ une procédure `som_prod_moy` permettant de calculer et retourner en un seul parcours du tableau le produit et la somme et la moyenne des éléments de ce tableau
5- Écrire en langage C/C++ le programme principal permettant de remplir un tableau **ne contenant aucun 0** puis d'afficher la somme, le produit et la moyenne de tous ses éléments en utilisant les sous-programmes précédents.