Université de Caen **Année 2007/2007**
UFR de Sciences **2ème session/septembre**

Licence L2
Programmation orientée objet
(Python II) : **EI31T**

Durée: 2H. Documents autorisés : notes personnelles.

Chaque candidat doit, au début de l'épreuve, porter son nom dans le coin de la copie qu'il cachera par collage après avoir été pointé. Il devra en outre porter son numéro de place sur chacune des copies, intercalaires, ou pièces annexées.

1. Construisez la classe d'objets **`M(a,b,c,d)`** qui simule/représente des matrices 2×2 : $\begin{pmatrix} a & b \\ c & d \end{pmatrix}$. Il faut definir le constructeur, la methode d'affichage en format **`[[a,b],[c,d]]`**, et quelques opérateurs arithmétiques : au moins l'addition et la multiplication des matrices, et la multiplication d'une matrice par un scalaire (par un nombre normal) ; ordre des arguments pourra alors être arbitraire, le programme doit accepter **`x*A`** et **`A*x`**, où **`x`** est un nombre, et **`A`** – une matrice. La construction d'autres méthodes et fonctions utiles peut vous apporter un bonus.

2. Écrire en Python un *générateur* **`powerset(l)`**, qui accepte une liste **`l`** comme argument, et qui retourne (par **`yield`**, bien sûr) une par une, la séquence de toutes les sous-listes de la liste **`l`**. Donc, **`powerset([1,2,3])`** doit engendrer (l'ordre n'est pas spécifié) les listes : **`[], [1], [2], [3], [1,2], [1,3], [2,3], [1,2,3]`**, au total $8 = 2^3$. L'algorithme est récursif, et il est basé sur une simple observation. Pour tout élément de la liste-argument, on *peut accepter* cet élément, et aussi *ne le pas accepter*. Donc, on construit le générateur secondaire pour la queue de la liste, et on génère deux solutions pour chacune offerte par ce générateur secondaire : sans ou avec la tête de la liste. *(Cet exercice vaut plus que les autres).*

3. Le développement de la fonction sinus en série de Taylor est :

$$\sin(x) = x - \frac{x^3}{3!} + \frac{x^5}{5!} - \frac{x^7}{7!} + \frac{x^9}{9!} - \dots$$

   Construire une fonction **`monsin(x)`** qui prend un argument réel **`x`**, qui réduit cet argument modulo 2π s'il est trop grand, et qui calcule la somme jusqu'à la convergence : quand l'élément suivant (sa valeur absolue) de la somme est plus petit que, disons, 0.00001, l'itération s'arrête. Le programme doit être écrit avec un minimum de professionalisme, relativement efficace. Une répétition inutile des calculs sera pénalisée.

Attention : **Je vous prie d'écrire de manière lisible, tout texte difficile à déchiffrer risque d'être rejeté. Ne jamais copiez des morceaux de code inutile, tout code hors sujet (surtout : tout le code des notes du cours, etc.) sera *pénalisé !* Évitez toute discussion qualitative (« philosophique ») inutile. Par contre, *commentez votre code,* sinon il risque d'être peu compréhensible.**