Des éléments de syntaxe Python, et en particulier l'usage du module numpy, sont donnés en annexe . Dans tout ce qui suit, les variables n, p, A, M, i, j et c vérifient les conditions suivantes qui ne seront pas rappelées à chaque question :

— n et p sont des entiers naturels tels que $p \geq n \geq 2$;

— A est une matrice carrée à n lignes inversible ;

— M est une matrice à n lignes et p colonnes telle que la sous-matrice carrée constituée des n premières colonnes de M est inversible ;

— i et j sont des entiers tels que $0 \leq i \leq n-1$ et $0 \leq j \leq n-1$;

— c est un réel non nul.

On note $L_i \leftarrow L_i + cL_j$ l'opération qui ajoute à la ligne i d'une matrice la ligne j multipliée par c.

1. Soit la fonction initialisation :

```
def initialisation(A):
    n = np.shape(A)[0]
    mat = np.zeros((n,2*n))
    for i in range(0, n):
        for j in range(0, n):
            mat[i,j] = A[i,j]
    return(mat)
```

Pour chacune des affirmations suivantes, indiquer si elle est vraie ou fausse, en justifiant. L'appel initialisation (A) renvoie :

(a) une matrice rectangulaire à n lignes et $2n$ colonnes remplie de zéros ;

(b) une matrice de même taille que A ;

(c) une erreur au niveau d'un range ;

(d) une matrice rectangulaire telle que les n premières colonnes correspondent aux n colonnes de A, et les autres colonnes sont nulles.

2. Les trois fonctions multip, ajout et permut suivantes ne renvoient rien : elles modifient les matrices auxquelles elles s'appliquent.

(a) Que réalise la fonction multip ?

```
def multip(M, i, c):
    p = np.shape(M)[1]
    for k in range(0, p):
        M[i,k] = c*M[i,k]
```

(b) Compléter la fonction ajout, afin qu'elle effectue l'opération $L_i \leftarrow L_i + cL_j$.

```
def ajout(M, i, j, c):
    p = np.shape(M)[1]
    for k in range(0, p):
        _____ ligne(s) a completer _____
```

(c) Écrire une fonction permut prenant pour argument M, i et j, et qui modifie M en échangeant les valeurs des lignes i et j.

Dans la suite du sujet, l'expression "opération élémentaire sur les lignes" fera référence à l'utilisation de permut, multip ou ajout.

3. Soit la colonne numéro j dans la matrice M. On cherche le numéro r d'une ligne où est situé le plus grand coefficient (en valeur absolue) de cette colonne parmi les lignes j à $n-1$. Autrement dit, r vérifie :

$$|A[r,j]| = \max\{|A[i,j]| \text{ pour } i \text{ tel que } j \leq i \leq n-1\}.$$

Écrire une fonction `rang_pivot` prenant pour argument M et j, et qui renvoie cette valeur de r.

Lorsqu'il y a plusieurs réponses possibles pour r, dire (avec justification) si l'algorithme renvoie le plus petit r, le plus grand r ou un autre choix.

(L'utilisation d'une commande max déjà programmée dans Python est bien sûr proscrite.)

4. Soit la fonction mystere :

```
def mystere(M):
    n = np.shape(M)[0]
    for j in range(0, n):
        r = rang_pivot(M, j)
        permut(M, r, j)
        for k in range(j+1, n):
            ajout(M, k, j, -M[k,j]/M[j,j])
        print(M)
```

(a) On considère dans cette question l'algorithme mystere appliqué à la matrice

$$M_1 = \begin{pmatrix} 3 & 2 & 2 \\ -6 & 0 & 12 \\ 1 & 1 & -3 \end{pmatrix}$$

Indiquer combien de fois la ligne print (M) est exécutée ainsi que les différentes valeurs qu'elle affiche

(b) De façon générale, que réalise cet algorithme ?

5. On considère la fonction reduire :

```
def reduire(M):
    n = np.shape(M)[0]
    mystere(M)
    for i in range(0, n):
        multip(M, i, 1/M[i,i])
    #Les lignes suivantes sont \'a compl\'eter :
        ____________________________
```

Compléter la fonction afin que la portion de code manquante effectue les opérations élémentaires suivantes sur les lignes :

pour j prenant les valeurs $n-1, n-2, \ldots, 1$, faire :

pour k prenant les valeurs $j-1, j-2, \ldots, 0$, faire :

$$\mathrm{L_k} \leftarrow \mathrm{L_k} - \mathrm{M[k,j]L_j}$$

Indiquer ce que réalise cette fonction.

Annexe

On considère que le module numpy, permettant de manipuler des tableaux à deux dimensions, est importé via import numpy as np. Pour une matrice M à n lignes et p colonnes, les indices vont de 0 à $n-1$ pour les lignes et de 0 à $p-1$ pour les colonnes.

Python	Interprétation
abs(x)	Valeur absolue du nombre x
M[i, j]	Coefficient d'indice (i, j) de la matrice M
np· zeros ((n, p))	Matrice à n lignes et p colonnes remplie de zéros
T = np.shape(M)	Dimensions de la matrice M
T[0] ou np.shape (M)[0]	Nombre de lignes
T[1] ou np.shape (M)[1]	Nombre de colonnes
M[a : b, c : d]	Matrice extraite de M constituée des lignes a à $b-1$ et des colonnes c à $d-1$: si a (resp. c) n'est pas précisé, l'extraction commence à la première ligne (resp. colonne) si b (resp. d) n'est pas précisé, l'extraction finit à la dernière ligne (resp. colonne) incluse