```
def Combinaison(A, i, j, coeff) :
....for k in range(len(A[0])) :
........A[i][k] += coeff*A[j][k]
```

```
def PermuteLignes(A, i, j) :
....for k in range(len(A[0])) :
........A[i][k], A[j][k] = A[j][k], A[i][k]
```

```
def CherchePivot(A, Col) :
....MeilleureLigne = Col
....for Lig in range(Col+1, len(A)) :
........if abs(A[Lig][Col]) > abs(A[MeilleureLigne][Col]) :
............MeilleureLigne = Lig
....return(MeilleureLigne)
```

```
def Pivot(Matrice, SecondMembre) :
....A = [Ligne[:]  for Ligne in Matrice]
....Y = [Ligne[:]  for Ligne in SecondMembre]
....for i in range(len(A)) :
........BestLigne = CherchePivot(A, i) :
........if BestLigne > i :
............PermuteLignes(A, i, BestLigne)
............PermuteLignes(Y, i, BestLigne)
........for j in range(i+1, n) :
............coeff = -A[j][i]/float(A[i][i])
............Combinaison(A, j, i, coeff)
............Combinaison(Y, j, i, coeff)
....Sol = [float(0) for i in range(n)]
....for k in range(n) :
........Lig = len(A)-k-1
........Sol[i] = (Y[i][0]-sum(A[i][j]*Sol[j] for j in range(i+1, n)) / A[i][i]
....return(Sol)
```

```
def delRowCol(M, i, k) :
....MM = []
....for ii in range(len(M)) :
........LL = []
........if ii != i :
............for kk in range(len(M[ii])) :
................if kk != k :
....................LL.append(M[ii][kk])
........MM.append(LL)
....return(MM)

def det(M) :
....if len(M)==0 :
........return(1)
....S, signe = 0, 1
....for i in range(len(M)) :
........S+=signe*M[i][0]*det(delRowCol(M,i,0))
........signe=-signe
....return(S)
```