Problema: Determinante de una matriz cuadrada (triangularizando)

In [1]:
```
m=[[2,1,3],[4,2,3],[2,3,2]]
```

In [2]:
```
def multiplicaFila(m,f,e):
    n=len(m)
    for c in range(n):
        m[f][c]=m[f][c]*e
```

In [3]:
```
multiplicaFila(m,0,1./2)
```

In [4]:
```
m
```

Out[4]:
```
[[1.0, 0.5, 1.5], [4, 2, 3], [2, 3, 2]]
```

In [5]:
```
def muestra(m):
    for f in m:
        print '[',
        for e in f:
            print "{:.2f}".format(e).rjust(6),
        print ']'
```

In [6]:
```
muestra(m)
```

```
[   1.00   0.50   1.50 ]
[   4.00   2.00   3.00 ]
[   2.00   3.00   2.00 ]
```

In [7]:
```
def combinacion(m,i,j,e):
    n=len(m)
    for c in range(n):
        m[j][c]=m[j][c]+e*m[i][c]
```

In [8]:
```
combinacion(m,0,1,-m[1][0])
```

In [9]:
```
muestra(m)
```

```
[   1.00   0.50   1.50 ]
[   0.00   0.00  -3.00 ]
[   2.00   3.00   2.00 ]
```

In [10]:
```
combinacion(m,0,2,-m[2][0])
```

In [11]:
```
muestra(m)
```

```
[   1.00   0.50   1.50 ]
[   0.00   0.00  -3.00 ]
[   0.00   2.00  -1.00 ]
```

In [12]:
```
def intercambiaFilas(m,i,j):
    m[i],m[j] = m[j],m[i]
```

Intercambiar columnas sería más difícil

In [13]:
```
intercambiaFilas(m,1,2)
muestra(m)
```

```
[   1.00   0.50   1.50 ]
[   0.00   2.00  -1.00 ]
[   0.00   0.00  -3.00 ]
```

In [14]:
```
multiplicaFila(m,1,1./m[1][1])
```

In [15]:
```
muestra(m)
```

```
[   1.00   0.50   1.50 ]
[   0.00   1.00  -0.50 ]
[   0.00   0.00  -3.00 ]
```

In [16]:
```
multiplicaFila(m,2,1./m[2][2])
```

In [17]:
```
muestra(m)
```

```
[   1.00   0.50   1.50 ]
[   0.00   1.00  -0.50 ]
[  -0.00  -0.00   1.00 ]
```

In [18]:
```
def determinante(m):
    n=len(m)

    det=1
    for i in range(len(m)):
        j=primeroNoNulo(m,i)
        if j==len(m):
            return 0
        if i!=j:
           det=-1*det
        intercambiaFilas(m,i,j)
        det=det*m[i][i]
        multiplicaFila(m,i,1./m[i][i])
        for k in range(i+1,n):
            combinacion(m,i,k,-m[k][i])
    return det

def primeroNoNulo(m,i):
    result=i
    while result<len(m) and m[result][i]==0:
        result=result+1
    return result
```

In [19]:
```
m=[[2,1,3],[4,2,6],[2,3,2]]
```

In [20]:
```
determinante(m)
```

Out[20]:
```
0
```

In [21]:
```
m
```

Out[21]:
```
[[1.0, 0.5, 1.5], [0.0, 1.0, -0.5], [0.0, 0.0, 0.0]]
```

Para no modificar `m` en la función debemos crear una copia y trabajar en ella

In [22]:
```
def copy(m):
    result=[]
    for f in m:
        result.append(f[:])
    return result
```

In [23]:
```
def determinante(m1):
    m=copy(m1)
    n=len(m)
    det=1
    for i in range(len(m)):
        j=primeroNoNulo(m,i)
        if j==len(m):
            return 0
        det=-1*det
        intercambiaFilas(m,i,j)
        det=det*m[i][i]
        multiplicaFila(m,i,1./m[i][i])
        for k in range(i+1,n):
            combinacion(m,i,k,-m[k][i])
    return det
```

In [24]:
```
m=[[2,1,3],[4,2,3],[2,3,2]]
```

In [25]:
```
determinante(m)
```

Out[25]:
```
12.0
```

In [26]:
```
m
```

Out[26]:
```
[[2, 1, 3], [4, 2, 3], [2, 3, 2]]
```

Tenemos todos los ingredientes para ser capaces de calcular la inversa

Sólo necesitamos la matriz identidad del mismo tamaño que `m`

In [27]:
```
def unitMatrix(n):
    result=[]
    for f in range(n):
        fila=[0]*n
        fila[f]=1
        result.append(fila)
    return result
```

In [28]:
```
unitMatrix(3)
```

Out[28]:
```
[[1, 0, 0], [0, 1, 0], [0, 0, 1]]
```

In [29]:
```
def inversa(m1):
    m=copy(m1)
    n=len(m)
    inversa=unitMatrix(n)
    det=1
    for i in range(len(m)):
        j=primeroNoNulo(m,i)
        if j==len(m):
            raise Exception('La matriz no es invertible')
        det=-1*det
        intercambiaFilas(m,i,j)
        intercambiaFilas(inversa,i,j)
        multiplicaFila(inversa,i,1./m[i][i])
        multiplicaFila(m,i,1./m[i][i])
        for k in range(0,n):
            if k!=i:
                combinacion(inversa,i,k,-m[k][i])
                combinacion(m,i,k,-m[k][i])
    return inversa
```

In [30]:
```
muestra(inversa(m))
```

```
[  -0.42   0.58  -0.25 ]
[  -0.17  -0.17   0.50 ]
[   0.67  -0.33  -0.00 ]
```

In [31]:
```
def matrizNula(n,m):
    result=[]
    for i in range(n):
        result.append([0]*m)
    return result


def mult(a,b):
    n,m=len(a),len(b[0])
    prod=matrizNula(n,m)
    for i in range(n):
        for j in range(m):
            for k in range(len(a[0])):
                prod[i][j]=prod[i][j]+a[i][k]*b[k][j]
    return prod
```

In [32]:
```
muestra(mult(m,inversa(m)))
```

```
[   1.00  -0.00   0.00 ]
[   0.00   1.00   0.00 ]
[   0.00  -0.00   1.00 ]
```

In [32]: