Problema 2

```
>>> b = numpy.array([23, 0, 0, 0])
>>> A = numpy.array([[1, 1, 1, 1],[1, 0, 1.25, 0],[0, 0.375, 0, 1],[0, 0, -1, 1]])
eliminación de Gauss
>>> x = numericoUtils.elimGauss(A, b, v = True)
Matriz aumentada:
[[ 1.     1.     1.     1.    23.   ]
 [ 1.     0.     1.25   0.     0.   ]
 [ 0.     0.375  0.     1.     0.   ]
 [ 0.     0.    -1.     1.     0.   ]]
alpha_0:
[[0.]
 [1.]
 [0.]
 [0.]]
e_0:
[1. 0. 0. 0.]
alpha_0 * e_0:
[[0. 0. 0. 0.]
 [1. 0. 0. 0.]
 [0. 0. 0. 0.]
 [0. 0. 0. 0.]]
L_0:
[[ 1.  0.  0.  0.]
 [-1.  1.  0.  0.]
 [ 0.  0.  1.  0.]
 [ 0.  0.  0.  1.]]
L_0 * a_b:
[[  1.      1.      1.      1.     23.   ]
 [  0.     -1.      0.25   -1.    -23.   ]
 [  0.      0.375   0.      1.      0.   ]
 [  0.      0.     -1.      1.      0.   ]]
alpha_1:
[[ 0.   ]
 [ 0.   ]
 [-0.375]
 [-0.   ]]
e_1:
[0. 1. 0. 0.]
alpha_1 * e_1:
[[ 0.     0.     0.     0.   ]
 [ 0.     0.     0.     0.   ]
 [-0.    -0.375 -0.    -0.   ]
 [-0.    -0.    -0.    -0.   ]]
L_1:
[[1.    0.    0.    0.   ]
 [0.    1.    0.    0.   ]
 [0.    0.375 1.    0.   ]
 [0.    0.    0.    1.   ]]
L_1 * a_b:
[[  1.        1.        1.        1.       23.     ]
 [  0.       -1.        0.25     -1.      -23.     ]
```

```
 [  0.        0.        0.09375   0.625    -8.625  ]
 [  0.        0.       -1.        1.        0.     ]]
alpha_2:
[[  0.        ]
 [  0.        ]
 [  0.        ]
 [-10.66666667]]
e_2:
[0. 0. 1. 0.]
alpha_2 * e_2:
[[  0.          0.          0.          0.        ]
 [  0.          0.          0.          0.        ]
 [  0.          0.          0.          0.        ]
 [ -0.         -0.        -10.66666667  -0.        ]]
L_2:
[[ 1.          0.          0.          0.        ]
 [ 0.          1.          0.          0.        ]
 [ 0.          0.          1.          0.        ]
 [ 0.          0.         10.66666667  1.        ]]
L_2 * a_b:
[[  1.          1.          1.          1.         23.        ]
 [  0.         -1.          0.25       -1.        -23.        ]
 [  0.          0.          0.09375     0.625     -8.625     ]
 [  0.          0.          0.          7.66666667 -92.        ]]
alpha_3:
[[0.]
 [0.]
 [0.]
 [0.]]
e_3:
[0. 0. 0. 1.]
alpha_3 * e_3:
[[0. 0. 0. 0.]
 [0. 0. 0. 0.]
 [0. 0. 0. 0.]
 [0. 0. 0. 0.]]
L_3:
[[1. 0. 0. 0.]
 [0. 1. 0. 0.]
 [0. 0. 1. 0.]
 [0. 0. 0. 1.]]
L_3 * a_b:
[[  1.          1.          1.          1.         23.        ]
 [  0.         -1.          0.25       -1.        -23.        ]
 [  0.          0.          0.09375     0.625     -8.625     ]
 [  0.          0.          0.          7.66666667 -92.        ]]
a_b final:
 [[  1.          1.          1.          1.         23.        ]
 [  0.         -1.          0.25       -1.        -23.        ]
 [  0.          0.          0.09375     0.625     -8.625     ]
 [  0.          0.          0.          7.66666667 -92.        ]]
>>> print(x) # Vector solución
```

```
[ 15.  32. -12. -12.]
>>> numericoUtils.Cond(A, b, x, x, v = True)
Vector error:
[0. 0. 0. 0.]
Vector error residual:
[ 0.00000000e+00  5.32907052e-15 -1.77635684e-15 -1.06581410e-14]
False

GaussJordan
>>> xgj = numericoUtils.gaussJordan(A, b, v = True)
Matriz aumentada:
[[ 1.     1.     1.     1.    23.   ]
 [ 1.     0.     1.25   0.     0.   ]
 [ 0.     0.375  0.     1.     0.   ]
 [ 0.     0.    -1.     1.     0.   ]]
alpha_0:
[[0.]
 [1.]
 [0.]
 [0.]]
e_0:
[1. 0. 0. 0.]
alpha_0 * e_0:
[[0. 0. 0. 0.]
 [1. 0. 0. 0.]
 [0. 0. 0. 0.]
 [0. 0. 0. 0.]]
T_0:
[[ 1.  0.  0.  0.]
 [-1.  1.  0.  0.]
 [ 0.  0.  1.  0.]
 [ 0.  0.  0.  1.]]
T_0 * a_b:
[[  1.      1.      1.      1.     23.   ]
 [  0.     -1.      0.25   -1.    -23.   ]
 [  0.      0.375   0.      1.      0.   ]
 [  0.      0.     -1.      1.      0.   ]]
alpha_1:
[[-1.   ]
 [ 2.   ]
 [-0.375]
 [-0.   ]]
e_1:
[0. 1. 0. 0.]
alpha_1 * e_1:
[[-0.    -1.    -0.    -0.   ]
 [ 0.     2.     0.     0.   ]
 [-0.    -0.375 -0.    -0.   ]
 [-0.    -0.    -0.    -0.   ]]
T_1:
[[ 1.     1.     0.     0.   ]
 [ 0.    -1.     0.     0.   ]
```

```
 [ 0.     0.375  1.     0.   ]
 [ 0.     0.     0.     1.   ]]
T_1 * a_b:
[[ 1.       0.       1.25     0.       0.     ]
 [ 0.       1.      -0.25     1.      23.     ]
 [ 0.       0.       0.09375  0.625   -8.625  ]
 [ 0.       0.      -1.       1.       0.     ]]
alpha_2:
[[ 13.33333333]
 [ -2.66666667]
 [ -9.66666667]
 [-10.66666667]]
e_2:
[0. 0. 1. 0.]
alpha_2 * e_2:
[[  0.          0.         13.33333333   0.        ]
 [ -0.         -0.         -2.66666667  -0.        ]
 [ -0.         -0.         -9.66666667  -0.        ]
 [ -0.         -0.        -10.66666667  -0.        ]]
T_2:
[[  1.          0.        -13.33333333   0.        ]
 [  0.          1.          2.66666667   0.        ]
 [  0.          0.         10.66666667   0.        ]
 [  0.          0.         10.66666667   1.        ]]
T_2 * a_b:
[[ 1.00000000e+00  0.00000000e+00 -5.55111512e-17 -8.33333333e+00
   1.15000000e+02]
 [ 0.00000000e+00  1.00000000e+00 -1.38777878e-17  2.66666667e+00
   1.27675648e-15]
 [ 0.00000000e+00  0.00000000e+00  1.00000000e+00  6.66666667e+00
  -9.20000000e+01]
 [ 0.00000000e+00  0.00000000e+00  0.00000000e+00  7.66666667e+00
  -9.20000000e+01]]
alpha_3:
[[-1.08695652]
 [ 0.34782609]
 [ 0.86956522]
 [ 0.86956522]]
e_3:
[0. 0. 0. 1.]
alpha_3 * e_3:
[[-0.         -0.         -0.         -1.08695652]
 [ 0.          0.          0.          0.34782609]
 [ 0.          0.          0.          0.86956522]
 [ 0.          0.          0.          0.86956522]]
T_3:
[[ 1.          0.          0.          1.08695652]
 [ 0.          1.          0.         -0.34782609]
 [ 0.          0.          1.         -0.86956522]
 [ 0.          0.          0.          0.13043478]]
T_3 * a_b:
[[ 1.00000000e+00  0.00000000e+00 -5.55111512e-17  2.25262643e-17
```

```
   1.50000000e+01]
 [ 0.00000000e+00  1.00000000e+00 -1.38777878e-17  1.31939548e-16
   3.20000000e+01]
 [ 0.00000000e+00  0.00000000e+00  1.00000000e+00  1.07804265e-16
  -1.20000000e+01]
 [ 0.00000000e+00  0.00000000e+00  0.00000000e+00  1.00000000e+00
  -1.20000000e+01]]
a_b final:
 [[ 1.00000000e+00  0.00000000e+00 -5.55111512e-17  2.25262643e-17
   1.50000000e+01]
 [ 0.00000000e+00  1.00000000e+00 -1.38777878e-17  1.31939548e-16
   3.20000000e+01]
 [ 0.00000000e+00  0.00000000e+00  1.00000000e+00  1.07804265e-16
  -1.20000000e+01]
 [ 0.00000000e+00  0.00000000e+00  0.00000000e+00  1.00000000e+00
  -1.20000000e+01]]
>>> print(xgj)
[ 15.  32. -12. -12.]
```

Problema 3

```
>>> A = numpy.array([[1, 1, 1],[1, 2, 4],[1, 4.85, 23.5225],[1, 6.75, 45.5625],[1, 9, 81]])
>>> A
array([[ 1.    ,  1.    ,  1.    ],
       [ 1.    ,  2.    ,  4.    ],
       [ 1.    ,  4.85  , 23.5225],
       [ 1.    ,  6.75  , 45.5625],
       [ 1.    ,  9.    , 81.    ]])
>>> b = [100, 82.9, 10, 24.4, 100]
>>> b = numpy.array(b)
>>> b
array([100. ,  82.9,  10. ,  24.4, 100. ])

>>> Ac = numpy.matmul(A.T, A)
>>> bc = numpy.matmul(A.T, b)

>>> print(Ac)
[[5.00000000e+00 2.36000000e+01 1.55085000e+02]
 [2.36000000e+01 1.55085000e+02 1.15963100e+03]
 [1.55085000e+02 1.15963100e+03 9.20724941e+03]]
>>> print(bc)
[ 317.3  1379.   9878.55]
>>> x = LU.L1U_fact(Ac, bc, v = True)
Matriz aumentada:
[[5.00000000e+00 2.36000000e+01 1.55085000e+02 3.17300000e+02]
 [2.36000000e+01 1.55085000e+02 1.15963100e+03 1.37900000e+03]
 [1.55085000e+02 1.15963100e+03 9.20724941e+03 9.87855000e+03]]
alpha_0:
[[ 0.   ]
 [ 4.72 ]
 [31.017]]
e_0:
```

```
[1. 0. 0.]
alpha_0 * e_0:
[[ 0.     0.     0.   ]
 [ 4.72   0.     0.   ]
 [31.017  0.     0.   ]]
L_0:
[[  1.      0.      0.   ]
 [ -4.72    1.      0.   ]
 [-31.017   0.      1.   ]]
L_0 * a_b:
[[   5.         23.6        155.085      317.3      ]
 [   0.         43.693      427.6298    -118.656    ]
 [   0.        427.6298    4396.9779675   36.8559   ]]
alpha_1:
[[0.        ]
 [0.        ]
 [9.78714668]]
e_1:
[0. 1. 0.]
alpha_1 * e_1:
[[0.         0.         0.        ]
 [0.         0.         0.        ]
 [0.         9.78714668 0.        ]]
L_1:
[[ 1.          0.          0.        ]
 [ 0.          1.          0.        ]
 [ 0.         -9.78714668  1.        ]]
L_1 * a_b:
[[   5.         23.6        155.085       317.3       ]
 [   0.         43.693       427.6298     -118.656     ]
 [   0.          0.          211.70238908 1198.15957676]]
alpha_2:
[[0.]
 [0.]
 [0.]]
e_2:
[0. 0. 1.]
alpha_2 * e_2:
[[0. 0. 0.]
 [0. 0. 0.]
 [0. 0. 0.]]
L_2:
[[1. 0. 0.]
 [0. 1. 0.]
 [0. 0. 1.]]
L_2 * a_b:
[[   5.         23.6        155.085       317.3       ]
 [   0.         43.693       427.6298     -118.656     ]
 [   0.          0.          211.70238908 1198.15957676]]
a_b final:
 [[   5.         23.6        155.085       317.3       ]
 [   0.         43.693       427.6298     -118.656     ]
```

```
 [   0.           0.         211.70238908 1198.15957676]]
L:
[[ 1.00000000e+00 -2.61442608e-17  0.00000000e+00]
 [ 4.72000000e+00  1.00000000e+00 -0.00000000e+00]
 [ 3.10170000e+01  9.78714668e+00  1.00000000e+00]]

U:
[[  5.         23.6       155.085     ]
 [  0.         43.693      427.6298    ]
 [  0.          0.        211.70238908]]
>>> print(x)
[162.18190335 -58.10741449   5.65964126]
```

Problema 4

```
>>> A = numpy.array([[4, -4], [1, 1]])
>>> b = numpy.array([-24, 12])
>>> l = numerico.Cholesky(A)
>>> l
array([[2.       , 0.       ],
       [0.5      , 0.8660254]])
```