Método de Crout

Sabemos que la factorización de Doolittle exige que la matriz L (de la factorización LU) que es triangular superior, tiene que tener 1 en la diagonal principal. Si aplicamos Doolittle a nuestra matriz A traspuesta, conseguiremos una factorización tipo
Crout tranpose(L.U)=transpose(U).tranpose(L)

```
a: matrix(
 [3,6,9],
 [1,4,11],
 [0,4,19]
);
```

(a) $\begin{bmatrix} 3 & 6 & 9 \\ 1 & 4 & 11 \\ 0 & 4 & 19 \end{bmatrix}$

```
l: matrix(
 [1,0,0],
 [0,1,0],
 [0,0,1]

);
```

(l) $\begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}$

```
u:ident(3);
```

(u) $\begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}$

```
/·u:matrix([0,0,0],[0,0,0],[0,0,0]);·/
```

(u) $\begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix}$

```
-->  b:matrix([1/2],[−2/3],[−3/4]);
```

(b) $\begin{bmatrix} \frac{1}{2} \\ -\frac{2}{3} \\ -\frac{3}{4} \end{bmatrix}$

Una vez tenemos todos los datos de partida, procedemos a construir el algoritmo para encontrar una factorización tipo Doolittle (LU), con 1 en la diagonal de L.

```
-->  for i:1 thru matrix_size(a)[1] do
                                    (for j:i thru matrix_size(a)[1] do
                                      u[i][j]:a[i][j]−sum(l[i][k]·u[k][j],k,1,i−1),
                                    for h:i+1 thru matrix_size(a)[1] do
                                      l[h][i]:1/u[i][i]·(a[h][i]−sum(l[h][k]·u[k][i],k,1,i
```

(%o47) *done*

```
-->  l;
```

(%o48) $\begin{bmatrix} 1 & 0 & 0 \\ \frac{1}{3} & 1 & 0 \\ 0 & 2 & 1 \end{bmatrix}$

```
-->  u;
```

(%o49) $\begin{bmatrix} 3 & 6 & 9 \\ 0 & 2 & 8 \\ 0 & 0 & 3 \end{bmatrix}$

```
-->  l.u;
```

(%o50) $\begin{bmatrix} 3 & 6 & 9 \\ 1 & 4 & 11 \\ 0 & 4 & 19 \end{bmatrix}$