```
> restart;with(LinearAlgebra):
>
```

Procédure dans le cas où il y a un carré

```
> sicarre:=proc(indice::integer,T0::Matrix)::list(Matrix);
  local T::Matrix,P::Matrix,i::integer,j::integer,n::integer;
  T:=T0;
  n:=RowDimension(T);
  P:=Matrix(n,n,0);
  for i from 1 to n do
    P[indice,i]:=T[indice,i]/T[indice,indice];
  od;

  T[[1..(indice-1),(indice+1)..n],[1..(indice-1),(indice+1)..n]]

  :=T[[1..(indice-1),(indice+1)..n],[1..(indice-1),(indice+1)..n]]

  -(1/T[indice,indice])*T[[1..indice-1,indice+1..n],indice].Transp
  ose(T[[1..indice-1,indice+1..n],indice]);

  for i from 1 to n do
    if (i<>indice) then
      T[indice,i]:=0;
      T[i,indice]:=0;
    fi;
  od;
  RETURN([T,P]);
  end proc:
```

Procédure dans le cas où il n'y a pas de facteur carré

```
> sipascarre:=proc(indice::list(integer),T0::Matrix)::list(Matrix)
  ;
  local
  T::Matrix,P::Matrix,B1::Vector,B2::Vector,B3::Matrix,i::integer,
  j::integer,n::integer,a;
  T:=T0;
  n:=RowDimension(T);
  P:=Matrix(n,n,0);
  B1:=2*T[[1..indice[1]-1,indice[1]+1..indice[2]-1,indice[2]+1..n]
  ,indice[1]];
  B2:=2*T[[1..indice[1]-1,indice[1]+1..indice[2]-1,indice[2]+1..n]
  ,indice[2]];
  #f_1 = t[1,2] en [1] + b2
  #f_2 = t[1,2]/t[1,2] en [2] + b1/t[1,2]
  B3:=B1.Transpose(B2);
  a:=2*T[indice[1],indice[2]];

  # matrice de changement de base:
  P[indice[1],[1..indice[1]-1,indice[1]+1..indice[2]-1,indice[2]+1
  ..n]]:=(B2+(1/a)*B1);
  P[indice[2],[1..indice[1]-1,indice[1]+1..indice[2]-1,indice[2]+1
  ..n]]:=(B2-(1/a)*B1);
  P[indice[1],[indice[1],indice[2]]]:=< a | 1 >;
  P[indice[2],[indice[1],indice[2]]]:=< a | -1 >;
```

```
# nouvelle forme quadratique:

T[[1..indice[1]-1,indice[1]+1..indice[2]-1,indice[2]+1..n],[1..i
ndice[1]-1,indice[1]+1..indice[2]-1,indice[2]+1..n]]

:=T[[1..indice[1]-1,indice[1]+1..indice[2]-1,indice[2]+1..n],[1.
.indice[1]-1,indice[1]+1..indice[2]-1,indice[2]+1..n]]
      -(1/a)*(1/2)*(Transpose(B3)+B3);
T[[1..indice[1]-1,indice[1]+1..n],indice[1]]:=Vector(n-1);
T[indice[1],[1..indice[1]-1,indice[1]+1..n]]:=Transpose(Vector(n
-1));
T[indice[1],indice[1]]:=1/4;
T[[1..indice[2]-1,indice[2]+1..n],indice[2]]:=Vector(n-1);
T[indice[2],[1..indice[2]-1,indice[2]+1..n]]:=Transpose(Vector(n
-1));
T[indice[2],indice[2]]:=-1/4;

RETURN([T,P]);
end proc:
```

décomposition de gauss

```
> GaussDec:=proc(T0::Matrix)::list(Matrix);
local
T::Matrix,P::Matrix,ListeResult1::list(Matrix),ListeResult2::lis
t(Matrix),i::integer,j::integer,n::integer;
T:=T0;
n:=RowDimension(T);
i:=1;
while (i <= n and T[i,i] = 0) do i:=i+1; od;
if (i <= n)
  then
    ListeResult1:=sicarre(i,T);
    T:=ListeResult1[1];
    P:=ListeResult1[2];

ListeResult2:=GaussDec(T[[1..(i-1),(i+1)..n],[1..(i-1),(i+1)..n]
]);
    T[[1..(i-1),(i+1)..n],[1..(i-1),(i+1)..n]]:=ListeResult2[1];
    P[[1..(i-1),(i+1)..n],[1..(i-1),(i+1)..n]]:=ListeResult2[2];
  else
    i:=1;
    j:=1;
    while (j <= n and T[i,j]=0) do
      j:=j+1;
      i:=1;
      while (i < j-1 and j <= n and T[i,j]=0) do i:=i+1; od;
    od;
    if (j <= n)
      then
        ListeResult1:=sipascarre([i,j],T);
        T:=ListeResult1[1];
        P:=ListeResult1[2];

ListeResult2:=GaussDec(T[[1..(i-1),(i+1)..(j-1),(j+1)..n],[1..(i
```

```
-1),(i+1)..(j-1),(j+1)..n]]);

T[[1..(i-1),(i+1)..(j-1),(j+1)..n],[1..(i-1),(i+1)..(j-1),(j+1).
.n]]:=ListeResult2[1];

P[[1..(i-1),(i+1)..(j-1),(j+1)..n],[1..(i-1),(i+1)..(j-1),(j+1).
.n]]:=ListeResult2[2];
      else
        # fournir le rangN:=n;
    fi;

fi;
RETURN([T,P]);
end proc:
> Teste:=proc(LR::list(Matrix))::Matrix;
Transpose(LR[2]).LR[1].LR[2];
RETURN(%);
end proc:
> Quad:=proc(Mat::Matrix);
expand(Transpose(Vector(RowDimension(Mat),symbol=x)).Mat.Vector(
RowDimension(Mat),symbol=x));
RETURN(%);
end proc:
>
> M:=RandomMatrix(7,7,generator=0..1,outputoptions=[shape=symmetri
c]);Rank(M);
```

$$M := \begin{bmatrix} 0 & 1 & 0 & 1 & 1 & 0 & 0 \\ 1 & 1 & 1 & 1 & 1 & 0 & 0 \\ 0 & 1 & 0 & 1 & 0 & 1 & 0 \\ 1 & 1 & 1 & 1 & 0 & 1 & 0 \\ 1 & 1 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 1 & 1 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix}$$

$$5$$

```
> LR:=GaussDec(copy(M));M:
```

$$LR := \left[\begin{bmatrix} -1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & -1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix}, \begin{bmatrix} 1 & 0 & 1 & 0 & 0 & 0 & 0 \\ 1 & 1 & 1 & 1 & 1 & 0 & 0 \\ 0 & 0 & 1 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & P & 0 & 0 & P \\ 0 & 0 & 1 & 1 & 1 & -1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & P & 0 & 0 & P \end{bmatrix}\right]$$

```
> M-Teste(LR);
```

$$\begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix}$$

```
> LR2:=GaussDec(M);
```

$$LR2 := \left[\begin{bmatrix} \frac{-1}{2} & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 2 \end{bmatrix}, \begin{bmatrix} 1 & -1 & 0 \\ 0 & P & 0 \\ \frac{1}{2} & \frac{1}{2} & 1 \end{bmatrix}\right]$$

```
> Teste(LR2)-M;
```

$$\begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix}$$

```
> M:=<<0|1|1>,<1|0|1>,<1|1|2>>;Rank(%);
```

$$M := \begin{bmatrix} 0 & 1 & 1 \\ 1 & 0 & 1 \\ 1 & 1 & 2 \end{bmatrix}$$

$$2$$

```
> Quad(M);
```

$$2\,x_1\,x_2 + 2\,x_1\,x_3 + 2\,x_2\,x_3 + 2\,x_3^{\;2}$$

```
> sicarre(3,M);
```

$$\left[\begin{bmatrix} \frac{-1}{2} & \frac{1}{2} & 0 \\ \frac{1}{2} & \frac{-1}{2} & 0 \\ 0 & 0 & 2 \end{bmatrix}, \begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ \frac{1}{2} & \frac{1}{2} & 1 \end{bmatrix}\right]$$

```
> sicarre(1,%[1]);
```

$$\left[\begin{bmatrix} \frac{-1}{2} & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 2 \end{bmatrix}, \begin{bmatrix} 1 & -1 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix}\right]$$

```
> LR2:=GaussDec(Matrix(5));
```

$$LR2 := \left[\begin{bmatrix} 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \end{bmatrix}, P\right]$$

```
>
```

$$T := \begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix}$$

```
>
> ?print
>
Error, reserved word `end` unexpected
```