```
> with(LinearAlgebra);
```

```
[&x, Add, Adjoint, BackwardSubstitute, BandMatrix, Basis, BezoutMatrix, BidiagonalForm,
 BilinearForm, CharacteristicMatrix, CharacteristicPolynomial, Column,
 ColumnDimension, ColumnOperation, ColumnSpace, CompanionMatrix,
 ConditionNumber, ConstantMatrix, ConstantVector, Copy, CreatePermutation,
 CrossProduct, DeleteColumn, DeleteRow, Determinant, Diagonal, DiagonalMatrix,
 Dimension, Dimensions, DotProduct, EigenConditionNumbers, Eigenvalues, Eigenvectors,
 Equal, ForwardSubstitute, FrobeniusForm, GaussianElimination, GenerateEquations,
 GenerateMatrix, Generic, GetResultDataType, GetResultShape, GivensRotationMatrix,
 GramSchmidt, HankelMatrix, HermiteForm, HermitianTranspose, HessenbergForm,
 HilbertMatrix, HouseholderMatrix, IdentityMatrix, IntersectionBasis, IsDefinite,
 IsOrthogonal, IsSimilar, IsUnitary, JordanBlockMatrix, JordanForm, KroneckerProduct,
 LA_Main, LUDecomposition, LeastSquares, LinearSolve, Map, Map2, MatrixAdd,
 MatrixExponential, MatrixFunction, MatrixInverse, MatrixMatrixMultiply, MatrixNorm,
 MatrixPower, MatrixScalarMultiply, MatrixVectorMultiply, MinimalPolynomial, Minor,
 Modular, Multiply, NoUserValue, Norm, Normalize, NullSpace, OuterProductMatrix,
 Permanent, Pivot, PopovForm, QRDecomposition, RandomMatrix, RandomVector, Rank,
 RationalCanonicalForm, ReducedRowEchelonForm, Row, RowDimension, RowOperation,
 RowSpace, ScalarMatrix, ScalarMultiply, ScalarVector, SchurForm, SingularValues,
 SmithForm, StronglyConnectedBlocks, SubMatrix, SubVector, SumBasis, SylvesterMatrix,
 ToeplitzMatrix, Trace, Transpose, TridiagonalForm, UnitVector, VandermondeMatrix,
 VectorAdd, VectorAngle, VectorMatrixMultiply, VectorNorm, VectorScalarMultiply,
 ZeroMatrix, ZeroVector, Zip]                                                    (1)
```

```
> Mat := proc(f, p)
  local n, M, i, j, P:
  n := degree(f):
  M := Matrix(n):
  for j from 1 to n do P := powmod(x^(j-1), p, f, x) mod p:
                       for i from 1 to n do M[i, j] := coeff(P, x, i-1) mod p :od:
  od:
  M := M - Matrix(n, shape = identity):
  return(M);
  end:
```

```
> f := x^9 + x^8 + x^6 + x^5 + x^3 + x + 1:
```

```
> M := Mat(f, 2);
```

(2)

$$M := \begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 1 \\ 0 & -1 & 0 & 0 & 0 & 0 & 1 & 0 & 1 \\ 0 & 1 & -1 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & -1 & 0 & 1 & 0 & 1 & 1 \\ 0 & 0 & 1 & 0 & -1 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & -1 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 1 & 0 & 0 & 0 \end{bmatrix} \quad (2)$$

```
> g := x^7 + x^2 + 1 : Mat(g, 5); q := x^6 + 2 x^5 + 7 x^4 + 10 x^3 + 12 x^2 + 8 x : N := Mat(q,
      3); NullSpace(N);
```

$$\begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & -1 & 0 & 1 & 3 & 3 & 1 \\ 0 & 0 & -1 & 0 & 0 & 0 & 1 \\ 0 & 0 & 4 & 1 & 2 & 4 & 0 \\ 0 & 0 & 0 & 0 & -1 & 4 & 4 \\ 0 & 1 & 4 & 1 & 4 & 0 & 4 \\ 0 & 0 & 0 & 0 & 1 & 2 & 0 \end{bmatrix}$$

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

$$\left\{ \begin{bmatrix} 0 \\ 0 \\ 0 \\ 0 \\ 1 \\ 0 \end{bmatrix}, \begin{bmatrix} 0 \\ 1 \\ 0 \\ 1 \\ 0 \\ 0 \end{bmatrix}, \begin{bmatrix} 1 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \end{bmatrix} \right\} \quad (3)$$

```
>
> base := proc(A, p)
  local N, n, S, Q, i, j, q :
  N := Nullspace(A) mod p : n := RowDimension(A) : S := NULL : q := nops(N) :
  for i from 1 to q do
```

```
                                    Q := 0 :for j from 1 to n do Q := Q + N[i](j, 1)·x^(j-1) mod p :
        od: S := S, Q:
                                od:
    S;
    end:
```

```
> base(M, 2);
```

$$1 \tag{4}$$

```
> facteur := proc(f, p)
    local M, S, a, Q, b, g, n, i :
    M := Mat(f, p) :
    S := base(M, p) :
    a := rand(0..p - 1) :
    Q := 0 : n := nops([S]) :
     for i from 1 to n do Q := Q + a( )·S[i] mod p :od:
    if degree(Q, x) < 1 then return(0) :
    else b := Gcd(f, Q) mod p :
                                    if degree(b, x) > 0 then return(b);
                                    else a := powmod(Q, (p - 1)/2, f, x) mod p :
                                                g := Gcd(a - 1, f) mod p : if g ≠ 1 and g ≠ f then
        return(g);
                                                                        else return(0) :
                                                                        fi:
                                fi:
    fi:
    end:
```

```
> f := x^8 + 4 x^4 + 3 x^2 + 1; facteur(x^6 + 2 x^5 + 7 x^4 + 10 x^3 + 12 x^2 + 8 x, 3);
```

$$f := x^8 + 4\,x^4 + 3\,x^2 + 1$$

$$x \tag{5}$$

```
> q := x^3 + 2 x^2 + 2 x + 1 : facteur(x^3 + 2 x^2 + 2 x + 1, 3);
```

$$x + 1 \tag{6}$$

```
> q := x^17 + x^13 + 4 x^6 + 3 x + 1 : gcd(q, q') mod 5;
```

$$1 \tag{7}$$

```
> facteur(q, 5);
```

$$x + 4 \tag{8}$$

```
>
> Berle := proc(f, p, k)
    local b, i :
    b := Gcd(f, f') mod p :
    if degree(b, x) > 0 then return(erreur);
```

```
    else i := 0 : b := 0 : while i < k and b = 0 do b := facteur( f, p ) : i := i + 1 :od:
    fi:
    if i = k then return(0)
    else return(b);
    fi:
    end:
```

```
> Berle( x^6 + 2 x^5 + 7 x^4 + 10 x^3 + 12 x^2 + 8 x, 3, 5 );
```

$$x^2 + x \tag{9}$$

```
> Berle( x^3 + x^2 + x + 2, 3, 5 );
```

$$0 \tag{10}$$

```
> decomp := proc( f, p, k )
  local g, S, u :
  S := [ ] :
  g := f:
  while Berle( g, p, k ) ≠ 0 do u := Berle( g, p, k ) : while Berle( u, p, k ) ≠ 0 do u := quo( u,
      Berle( u, p, k ), x ) mod p :od: S := [ op( S ), u ] : g := quo( g, u, x ) mod p :od:
  S := [ op( S ), g ];
  end:
```

```
> decomp( x^2 + 4, 5, 2 );
```

$$[x^2 + 4] \tag{11}$$

```
> decomp( x^6 + 2 x^5 + 7 x^4 + 10 x^3 + 12 x^2 + 8 x, 3, 5, 2 );
```

$$[2 + x^2 + x, x + 1, x, x^2 + 1] \tag{12}$$

```
> decomp( q, 5, 2 );
```

$$[x + 4, x^{15} + x^{13} + 2 x^{11} + 2 x^9 + 2 x^7 + 2 x^5 + 4 x^4 + 2 x^3 + 4 x^2 + 2 x + 4, x + 1] \tag{13}$$

```
>
```