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

[*&x*, *Add*, *Adjoint*, *BackwardSubstitute*, *BandMatrix*, *Basis*, *BezoutMatrix*, *BidiagonalForm*, *BilinearForm*, *CARE*, *CharacteristicMatrix*, *CharacteristicPolynomial*, *Column*, *ColumnDimension*, *ColumnOperation*, *ColumnSpace*, *CompanionMatrix*, *ConditionNumber*, *ConstantMatrix*, *ConstantVector*, *Copy*, *CreatePermutation*, *CrossProduct*, *DARE*, *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*, *LyapunovSolve*, *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*, *SylvesterSolve*, *ToeplitzMatrix*, *Trace*, *Transpose*, *TridiagonalForm*, *UnitVector*, *VandermondeMatrix*, *VectorAdd*, *VectorAngle*, *VectorMatrixMultiply*, *VectorNorm*, *VectorScalarMultiply*, *ZeroMatrix*, *ZeroVector*, *Zip*] **(1)**

```
> N:=10:
  v:=Vector(N);
```

$$v := \begin{bmatrix} 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \end{bmatrix} \quad \textbf{(2)}$$

```
> for j from 1 to N do v[j]:=j: od:
> A:= Matrix(N,shape=Vandermonde[v]);
```

(3)

$$A := \begin{bmatrix} 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & 2 & 4 & 8 & 16 & 32 & 64 & 128 & 256 & 512 \\ 1 & 3 & 9 & 27 & 81 & 243 & 729 & 2187 & 6561 & 19683 \\ 1 & 4 & 16 & 64 & 256 & 1024 & 4096 & 16384 & 65536 & 262144 \\ 1 & 5 & 25 & 125 & 625 & 3125 & 15625 & 78125 & 390625 & 1953125 \\ 1 & 6 & 36 & 216 & 1296 & 7776 & 46656 & 279936 & 1679616 & 10077696 \\ 1 & 7 & 49 & 343 & 2401 & 16807 & 117649 & 823543 & 5764801 & 40353607 \\ 1 & 8 & 64 & 512 & 4096 & 32768 & 262144 & 2097152 & 16777216 & 134217728 \\ 1 & 9 & 81 & 729 & 6561 & 59049 & 531441 & 4782969 & 43046721 & 387420489 \\ 1 & 10 & 100 & 1000 & 10000 & 100000 & 1000000 & 10000000 & 100000000 & 1000000000 \end{bmatrix} \quad (3)$$

```
> ConditionNumber(evalf(A));
```

$$3.306440917\ 10^{12} \quad (4)$$

$$Digits := 10 \quad (5)$$

```
> for j from 1 to N do v[j]:=evalf(exp(j*2*Pi*I/N)): od:
> A:= Matrix(N,shape=Vandermonde[v]):
> ConditionNumber(A);
```

$$10.00000000 \quad (6)$$

```
> for j from 1 to N do v[j]:=evalf(j): od:
> A:= Matrix(N,shape=Vandermonde[v]):
> ConditionNumber(A);
```

$$3.306440917\ 10^{12} \quad (7)$$

```
> for j from 1 to N do v[j]:=evalf(j/N): od:
> A:= Matrix(N,shape=Vandermonde[v]):
> ConditionNumber(A);
```

$$1.760000001\ 10^{8} \quad (8)$$