Vector functions	
Function signature	**Description**
vector = row_vector = create_vector vector(X), vector(K,X) col_vector(X) = vector(X,'c'), col_vector(K, X) = vector(K,X,'c')	The function vector(X) transforms a list X to a Vector. If a field or ring K is supplied as a the first parameter, the elements of X are pushed to elements of it. The parameter 'c' is to indicate that the constructed vector be a column vector.
vec(x,K) vec(x)	It returns vector(K,x). By default K = Zn(2)
eps(n,j) eps(m,n,j,k)	If n is a positive integer and 0≤j<n, eps(n,j) is the length n list whose j-th entry is 1 and all other entries are 0. If j is out of the indicated range, eps(n,j) is the length n list whose entries are all 0. Similarly, given positive integers m and n, and integers j and k such that 0≤j<m and 0≤k<n, eps(m,n,j,k) creates an m × n matrix with all entries 0 except 1 in the (j,k) entry. If either j or k is not in the stated range, the call returns the m × n null matrix. Examples: eps(4,1) ⇒ [0, 1, 0, 0] eps(3,4,2,2) ⇒ [[0 0 0 0] [0 0 0 0] [0 0 1 0]] :: Matrix[ZZ]
vector_append(x,a)	If x is a vector, this function is equivalent to vector(list(X)+[a])
error_vector(n,E)	If E is a list of pairs (j,v), j in 0..n-1, it creates a lenght n vector with values v at the positions j.
wt(x) hd(x,y)	If x is a list or vector, wt(x) is equivalent to len(support(x)). If y is another list or vector of the same length as x, hd(x,y) supplies the Hamming distance between x and y, which by definition is the cardinal of the set of indices j in the range of the lists/vectors such that xj ≠ yj. If both x and y are vectors, it is equivalent to wt(x-y).
cyclic_shift(x) cyclic_shifts(x)	Given x a vector [x1....xn], cyclic_shift(x) returns [xn,x1...xn-1]. cyclic_shifts(x) return a list with all the different cycles of x.
reverse(x)	If x = [x1,...,xn] is a list or a vector, it return the reverse element x'=[xn,....,x1]
support(x) pattern(x) histogram(x)	If x is a list or vector, support(x) yields the list of the indices i in the range of x such that xi≠0. The function pattern(x) and histogram returns a pair (s,v), with s = support(x) and v = vector([x[j] for j in s])
convolution(a,b,k) convolution(a,b)	If a and b are vectors or lists and k is an integer, this function returns the k-th coefficient of the convolution of a and b, namely the sum of the terms aj·bk-j for j = 0,...,k, with the conention that aj=0 if j≥len(x) and bk-j=0 if k-j≥len(b). Thus convolution(a,b,k) = 0 if k<0 or k≥len(a)+len(b). If a and b are lists, convolution(a,b) is the list [convolution(a,b,k) for k in range(n)], where n=len(a)+len(b)−1. For vectors, convolution(a,b) is equivalent to vector(convolution(list(a),list(b))).
geometric_series (x, n, s0) geometric_series (x, n)	The first call produces the vector [s0,s0·x,...,s0·x^(n−1)]. The second is defined as geometric_series(x,n,1), which therefore supplies the vector [1, x,...,x^(n−1)].

Vector functions	
Function signature	**Description**
flip(x) flip(x,k)	The first form transforms a list or a vector x into the list or vector whose components are 1-x[j], j in the range of x. For a binary list or vector, it concides with the result of swaping 0 and 1 (this explains the name given to the function). In the second form, k may be an index in the range of x, or a list/tuple/set of such indices and only the kth component, or the components with index in k, undergo the trasformation 1-x[j]. Examples: x = rd_vector(Zn(2),7) ⇒ [1, 0, 0, 1, 0, 0, 1] :: Vector[Z2] flip(x) ⇒ [1, 0, 1, 1, 0, 0, 1] :: Vector[Z2] flip(x) ⇒ [1, 0, 0, 1, 0, 0, 1] :: Vector[Z2] R = vector(Zn(5), list(range(9))) ⇒ [0,1,2,3,4,0,1,2,3] :: Vector(Z5) flip(R,(1,6)) ⇒ [0, 0, 2, 3, 4, 0, 0, 2, 3] :: Vector[Z5] flip(R) ⇒ [1, 1, 4, 3, 2, 1, 1, 4, 3] :: Vector[Z5]
invert_entries (a)	If a is a vector with no zero entries, it returns the vector [1/a1,...,1/an].
prod (x, y) dot (x, y)	If x and y are lists or vectors of the same length n, prod(x, y) constructs the vector [x1·y1,...,xn·yn] and dot(x, y) yields the sum x1·y1+⋯+xn·yn
rd_vector(K,n)	Given a domain K and a integer n, it returns a vector of n elements in K.
rd_nonzero_vector(K,n)	Given a domain K and a integer n, it returns a vector of n elements in K with not all elements equal to zero.
null_vector() null_vector(K)	It returns a vector of lenth 0. If a domain K is given, the null vector will have domain K.
subs_element(A,x,y)	Given a vector or matrix A, it changes all the entries equal to x by the elemnt y.
rd_error_vector (K, n, s) rd_error_vector (K, s)	If n and s are positive intgers, s≤n, and K is a field (or a ring), rd_error_vector(K, n, s) produces a vector in Kn of weight s whose non-zero positions and the corresponding values have been chosen randomly. The function rd_error_vector(K, s) is defined as rd_error_vector(K, q-1, s), where q is the cardinal of K.
bin_(x)	Given a list or a vector x, it returns a vector v equal to x changing the -1 elements by 0.
unbin_(x)	Given a list or a vector x, it returns a vector v equal to x changing the 0 elements by -1.
is_vector(v)	It returns if v is of type vector.
vector_resultant(v,w)	Given two vectors, it returns the resultant of the two vectors.
append_vector(*V)	Given a list of vectors it concatenates all the elements.