kan/sm1 operators reference

3.000224

1 Primitive Operators

1. QUIT

```
The bye or QUIT or quit statement terminates the kan/sm1.
```

2. Usage

```
The keyword <<Usage>> cannot be found.
Type in ? in order to see system dictionary.
```

3. [

```
[ and ] are used to construct an array.
The left bracket [ is an operator that leaves an object called
a mark on the stack. The interpreter puts more objects
until it encounters a right bracket, which creates
an array. So, it is possible to construct an array, for example, in the way
 [  1  2  [ 2 1 roll 3] ] print ==> [1 [2 3]]
```

4.]

```
[ and ] are used to construct an array.
The left bracket [ is an operator that leaves an object called
a mark on the stack. The interpreter puts more objects
until it encounters a right bracket, which creates
an array. So, it is possible to construct an array, for example, in the way
 [  1  2  [ 2 1 roll 3] ] print ==> [1 [2 3]]
```

5. add

```
<< obj1 obj2 add obj3 >>
obj3 is the sum of obj1 and obj2.
If obj1=f1/f2, obj2=g1/g2, then obj3 is given by (g2 f1 + f2 g1)/(f2 g2)
Example:  2 3 add ::  ==> 5
```

6. `aload`

```
<< [f1 f2 ... fn] aload f1 f2 ... fn [f1 ... fn] >>
The primitive aload decomposes an array into the set of the elements.
Example: [(x1+1) (23)] aload length ===> (x1+1) (23) 2
```

7. `bye`

```
The bye or QUIT or quit statement terminates the kan/sm1.
```

8. `cat_n`

```
<< s1 s2 ... sn n cat_n s1s2...sn >>
string s1, s2, ..., sn ; integer n ; string s1s2...sn
The primitive cat_n concatnates the strings(dollars) s1 ... sn.
Example: (ab c) (2) 2 cat_n ===> (ab c2)
```

9. `cclass`

```
<< [super-class-tag, o1, o2, ...] size [class-tag] cclass [class-tag, o1, o2, .... ] >>
```

10. `closefile`

```
<< fd closefile >>
file fd;
```

11. `coeff`

```
<< f v coeff [exponents coefficients] >>
poly f,v; list of integers exponents; list of poly coefficients;
The primitive coeff returns the array of exponents and
```

```
the array of coefficients of the polynomial f with respect to
the variable v.
The primitive coefficients have not yet removed for a compatibility with the old macro packages.
Note that the data type of each element of exponents is <Poly>(9) and it may cause a trouble when p>0.
cf. dc,  (interger) dc
C-functions: Kparts2, parts2
Example: [(x,y) ring_of_polynomials [[(x) 1 (y) 1]] weight_vector
           0] define_ring
         (x^2+ x y + x). (x). coeff print ;
  ===>  [ [ 2 , 1 ] , [1 , y+1] ]
```

12. coefficients

```
<< f v coeff [exponents coefficients] >>
poly f,v; list of integers exponents; list of poly coefficients;
The primitive coeff returns the array of exponents and
the array of coefficients of the polynomial f with respect to
the variable v.
The primitive coefficients have not yet removed for a compatibility with the old macro packages.
Note that the data type of each element of exponents is <Poly>(9) and it may cause a trouble when p>0.
cf. dc,  (interger) dc
C-functions: Kparts2, parts2
Example: [(x,y) ring_of_polynomials [[(x) 1 (y) 1]] weight_vector
           0] define_ring
         (x^2+ x y + x). (x). coeff print ;
  ===>  [ [ 2 , 1 ] , [1 , y+1] ]
```

13. copy

```
<< f1 f2 ... fn n copy f1 f2 ... fn f1 f2 ... fn >>
The primitive copy_n duplicates the objects f1 f2 ... fn.
integer n;
Example: (x1+1) 1 copy ===> (x1+1) (x1+1)
<<copy>> copies the values of the objects.  cf. dup
```

14. data_conversion

```
<< obj (type?) data_conversion type >>
<< obj (type??) data_conversion type >>
<< obj (key word) data_conversion obj >>
integer type; object obj;
Key words: null, integer, literal, dollar, string, poly, universalNumber,
           ring, numerator, denominator, double, cancel,
           map, orderMatrix, error,
 or [(class) (class-name)].
The primitive reports and converts the data type of obj.
Example: (2) (integer) data_conversion ===> 2
```

15. def

```
<< /name object def >>  or  << object /name set >>
The primitive def or set binds the object to the name
Example: /abc (23) def    abc :: ---> (23)
Example: (23) /abc set    abc :: ---> (23)
```

16. degree

```
<< f  v  degree  deg >>
poly f,v; integer deg;
deg is the degree of the polynomial f with respect to the variable v.
Example: (x^3 + x h + 1).  (x).  degree :: ---> 3
```

17. div

```
<< obj1  obj2 div obj3 >>
obj3 is obj1/obj2
cf. data_conversion (numerator, denominator, cancel)
```

18. dup

```
<< obj dup obj obj >>
cf. 1 copy. Dup copies the object by the pointer.
```

19. elimination_order

```
<< (v1,...,vm) elimination_order >>
The primitive returns the weight matrix where the variables
v1, ..., vm are most expensive terms.
cf. define_ring, define_qring.
```

20. eq

```
<< o1 o2 lt r1  >> or  <<  o1 o2 gt r1 >> or <<  o1 o2 gt r1 >>
integer r1;
These premitives compare objects o1 and o2.
[lt (less than), gt (greater than), eq (equal)]
The result is r1 which is integer object.
Here, 1 means true and 0 means false.
Example:  (abc) (bca) eq ::  ===> 0
```

21. `error`

```
<<error>> causes kan/sm1 error.
<< s error >>   string s;
s is printed as the error message.
```

22. `eval`

```
<< [$key$ arguments] eval >>
key :
```

23. `exec`

```
<< { executable array } exec >>
The primitive exec executes each primitive in the executable
array sequentially.
Example: {1 2 add print pop} exec ;
    ===> 3
```

24. `exit`

```
The keyword <<exit>> cannot be found.
Type in ? in order to see system dictionary.
```

25. `extension`

```
<< obj1 extension obj2 >>
array of object obj1; object obj2;
The first element of the obj1 should be the key word tag.
<< extension >> is used to install a new function to a system. cf.ext.c and plugin/
  [(parse) string] extension result-integer
  [(chattr) num literal] extension result-object
  [(chattrs) num ] extension result-object
  [(defaultPolyRing) num ] extension result-object
  [(flush) ] extension null
  [(getpid) ] extension result-integer
  [(getenv) envName] extension valueOfEnvName
  [(stat) fname] extension v
       v = [null,[(error no), p]] or [0, [size ]]
  [(forkExec) argList fdList sigblock] extension pid
  [(getchild)] extension listOfPid (generated by forkExec)
  [(keywords) ] extension array-of-names-of-primitives
```

```
  [(getUniqueFileName) path] extension newName
  [(outputObjectToFile) path obj] extension null
See also plugin-* in ??.
```

26. `file`

```
<< filename mode file fd >>
string filename, mode; file fd;
It opens the <<filename>> with the <<mode>>. <<fd>> is the
file descripter and is used in writestring and closefile.
If filename part is an integer, it calls fdopen() and returns the file descripter.
```

27. `for`

```
<<init inc limit {executable array} for >>
integer init inc limit;
Repeat the executable array.
Example: [ 1 1 3 {(a)} for] ::---> [1 (a) 2 (a) 3 (a)]
```

28. `gbext`

```
<< obj1 gbext obj2 >>
array of objects obj1;The first element of the obj1 should be the key word tag.
<< gbext >> is used to call auxiliary functions for g-basis computation.
  [(isReducible) poly1 poly2 ] gbext integer
  [(lcm) poly1 poly2] gbext poly
  [(grade) poly1 ] gbext integer
  [(mod) poly1 universalNumber] gbext poly
          poly = poly1 mod universalNumber where char=0 and
          poly and poly2 belong to a same ring.
  [(tomodp) poly1 ring] gbext poly, char(ring)>0.
          poly = poly1 mod char(ring) where poly belongs to ring.
  [(tomod0) poly1 ring] gbext poly, char(ring)=0.
  [(divByN) poly1 n(universalNumber)] gbext [qpoly rpoly]
     where poly1 = n*qpoly+rpoly.
     (see also cancelCoeff)
  [(isConstant) poly] gbext bool
  [(lcoeff) poly] gbext poly
  [(lmonom) poly] gbext poly
  [(schreyerSkelton) array_of_poly] gbext array
  [(toes) array_of_poly] gbext poly
  [(isOrdered) poly] gbext poly
```

29. `get`

```
<< [f0 f1 ... fn] k get fk >>
integer k;The primitive gets the k-th element of a given array.
Example: [1 4 3] 1 get :: ---> 4
```

30. goto

```
<< label goto >>
literal label;
Break the loops and jump to the <<literal>>.
The label <<literal>> has to be out of all the loops.
You cannot jump inside of the executable array { .... }.
Example: /point goto (skip me) message /point
```

31. groebner

```
<< [[f1 ... fn] [options]] groebner
   [[g1 ... gm] backward-transformation syzygy]>>
poly f1, ..., fn; poly g1, ..., gm;
optional return value: matrix of poly backward-transformation, syzygy;
Computation of the Groebner basis of f1,...,fn. The basis is {g1,...,gm}.
Options:  << (needBack), (needSyz), (countDown) number (StopDegree) number, (forceReduction)>>
Flags:<< [(ReduceLowerTerms) 1] system_variable >>
      << [(UseCriterion1) 0]    system_variable >>
      << [(UseCriterion2B) 0]   system_variable >>
      << [(Sugar) 0]            system_variable >>
      << [(Homogenize) 1]       system_variable >>
      << [(CheckHomogenization) 1] system_variable >>
      << [(Statistics) 1]       system_variable >>
      << [(KanGBmessage) 1]     system_variable >>
      << [(Verbose) 0]          system_variable >>
Example: [(x0,x1) ring_of_polynomials 0] define_ring
         [(x0^2+x1^2-h^2). (x0 x1 -4 h^2).] /ff set ;
         [ff] groebner /gg set ;
         gg ::
cf. homogenize, groebner_sugar, define_ring,
    ring_of_polynomials, ring_of_differential_operators.
```

32. gt

```
<< o1 o2 lt r1  >> or  <<  o1 o2 gt r1 >> or <<  o1 o2 gt r1 >>
integer r1;
These premitives compare objects o1 and o2.
[lt (less than), gt (greater than), eq (equal)]
The result is r1 which is integer object.
Here, 1 means true and 0 means false.
Example:  (abc) (bca) eq ::  ===> 0
```

33. homogenize

```
<< f homogenize g >>
poly f,g;
array of poly f,g;
```

34. idiv

```
<< a b idiv a/b >>
int a,b,a/b;
Example:  5 2 idiv ::  ===> 2
```

35. ifelse

```
<< condition { true case } { false case } ifelse >>
integer condition;
If condition is non-zero, then true-case will be executed,
else false-case will be executed.
```

36. index

```
<<  a_0 ... a_{n-1} n index a_0 ... a_{n-1} a_0 >>
integer n;
It takes a_0 from the stack.
```

37. init

```
<< f init h >>
<< f weight_vector init h >>
poly f,h; array of integer weight_vector;
h is the initial term of the polynomial f.
Example: (2 x^3 + 3 x + 1). init :: ---> 2 x^3
Example: [(x^3+1). (x h+ h^5).] {init} map ::
Example: (x+1). [0 1 1 1] init ::    cf. weightv
```

38. lc

```
<< obj lc leftCell >>
class leftCell;
leftCell is the left cell of the obj of which type must be class.
```

39. `length`

```
<< [a0 a1 ... an] length <n+1> >>
<< s length p >>
integer <n+1>; integer p; string s;
It returns the length of the argument. p is the length of the string.
```

40. `load`

```
<< /literal load obj >>
<<load>> gets the value of <<literal>>.
```

41. `loop`

```
<< {executable array} loop >>
cf. exit
```

42. `lt`

```
<< o1 o2 lt r1  >> or  <<  o1 o2 gt r1 >> or <<  o1 o2 gt r1 >>
integer r1;
These premitives compare objects o1 and o2.
[lt (less than), gt (greater than), eq (equal)]
The result is r1 which is integer object.
Here, 1 means true and 0 means false.
Example:  (abc) (bca) eq ::  ===> 0
```

43. `map`

```
<< [f1 ... fn] {code} map [f1 {code}exec ... fn {code}exec] >>
map applies {code} to each element in [f1 ... fn].
Example: [1 2 3] {1 add} map ===> [2 3 4]
```

44. `mpzext`

```
[(key) arguments] mpzext ans
It calls mpz functions.
arguments should be numbers(universalNumber).
Keys: gcd, tdiv_qr, cancel, powm, probab_prim_p, sqrt, and, ior, com
For details, see gmp-2.0.2/texinfo.tex.
```

45. mul

```
obj1 obj2 mul obj3
obj3 = obj1 * obj2
```

46. newcontext

```
<< name super newcontext class.Context >>
It creates new context and new user dictionary.
Ex. (mycontext) StandardContextp newcontext /nc set.
cf. setcontext.
```

47. newstack

```
<< size newstack class.OperandStack >>
cf. setstack, stdstack.
```

48. options

```
The keyword <<options>> cannot be found.
Type in ? in order to see system dictionary.
```

49. pop

```
<< obj pop >>
It removes the obj from the stack.
```

50. primmsg

```
<< { executable array } primmsg >>
The << executable array >> is executed in the PrimitiveContext.
cf. sendmsg, sendmsg2, supmsg2, system_variable
```

51. principal

```
<< obj principal obj >>  Get the principal part.
This is an old command. init should be used.
Example: (x Dx + Dy^2+ y Dx Dy). [(Dx) 1 (Dy) 1] weightv init
```

52. print

```
<< obj print >>
Print the object to stdout. cf. obj (string) dc
```

53. print_switch_status

```
<< print_switch_status >>
cf. switch_function
```

54. pstack

```
<< pstack >>
The operator prints the entire contents of the stack.
```

55. pushfile

```
<< filename pushfile sss >>
string filename, sss ;
Read the file and push the contents of file as the string.
The file will be search in the search path as run operator.
Example: (myfile.txt) pushfile /sss set
cf:  [(parse) sss] extension
```

56. put

```
<< [a0 a1 ... ap] i any put >>
<< [a0 a1 ... ap] multi-index any put >>
<< s i any put >>
integer i; string s; array of integers multi-index;
The operator put <<any>> at the place i.
Example: /a [1 2 3] def  a 2 (Hi) put a :: --->[1 2 (Hi)]
Example: /a [[1 2] [3 4]] def  a [0 1] 10 put a :: --->
                                              [[1 10] [3 4]]
```

57. quit

```
The bye or QUIT or quit statement terminates the kan/sm1.
```

58. rc

```
<< obj rc rightCell >>
rightCell is the right cell of the obj of which type must be class.
class rightCell;
```

59. reduction

```
<< f [g_1 g_2 ... g_m] reduction [h c0 syz input] >>
poly f, g1_1, ..., g_m, h, c0; array of poly syz, input;
h is the normal of f by {g1 .... fm}.
h = c0 f + \sum syz_i g_i
```

60. replace

```
<< f [[L1 R1] ... [Ln Rn]] replace g >>
poly f, L1, R1, ..., Ln, Rn, g;
<<replace>> applies the rules L1-->R1, ..., Ln-->Rn to transform f.
Example: (x^2+x+1). [[(x). (2).]] replace :: ---> 7
```

61. roll

```
<< f(n-1) ... f(0) n j roll f(j-1 mod n) f(j-2 mod n) ... f(j-n mod n) >>
integer n, j;
Example: [(a) (b) (c) 3 1 roll] ::  ==>[(c) (a) (b)]
```

62. run

```
<< filename run >>
string filename ;
Example. (restriction.sm1) run
File will be searched in the current directory, /usr/local/lib/sm1,
and the directory specified by the environment variable LOAD_SM1_PATH
Example. setenv LOAD_SM1_PATH ~/lib/sm1/
```

63. sendmsg

```
<< obj { executable array } sendmsg >>
When obj is in k-class (i.e. obj is of the form
 [<<class.Context>> , body ]), then the CurrentContextp is changed
to <<class.Context>> and the obj <<executable array>> will be executed.
If not, CurrentContextp is changed to the PrimitiveContextp and
the obj <<executable array>> will be executed.
cf. setcontext, newcontext, StandardContextp, system_variable,
    sendmsg2, primmsg, supmsg2.
```

64. sendmsg2

```
<< obj2 obj3 { executable array } sendmsg2 >>
Context is determined by obj2 if obj2 is class.context.
If not, it is determined by obj3 if obj2 is class.context.
cf. sendmsg.
```

65. set

```
<< /name object def >>  or  << object /name set >>
The primitive def or set binds the object to the name
Example: /abc (23) def    abc :: ---> (23)
Example: (23) /abc set    abc :: ---> (23)
```

66. set_order_by_matrix

```
<< order-matrix set_order_by_matrix >>
matrix of integers order-matrix;
cf. weight_vector, show_ring, elimination_order, switch_function
    system_variable
```

67. set_timer

```
cf. timer
```

68. set_up_ring@

```
<< [x-variables] [d-variables] [p c l m n cc ll mm nn next] order-matrix
   [(keyword) value (keyword) value ....]  set_up_ring@ >>
<<next>> is the optional argument. The last argument is also optional.
Keywords are mpMult, coeffient ring, valuation, characteristic,
             schreyer, ringName.

1.When [x[0] .... x[n-1]] [D[0] .... D[n-1]] is given as the lists
  of variables, D[0] is usually used as the variable for homogenization
  and x[n-1] is used for the variable for the graduation.
2.Order matrix should be given in the order x[n-1] ... x[0], D[n-1]...D[0]
3.0<=i<c : commutative; c<=i<l : q-difference;
  l<=i<m : difference(better not to use it); m<=i<n : differential;
4.Graduation variables :
  cc<=i<c : commutative; ll<=i<l : q-difference;
  mm<=i<m : difference;  nn<=i<n : differential;
  If you do not use graduation variables, set, for example, cc=c.
5.c-cc>0 or l-ll>0 or m-mm>0 or n-nn>0 must be held.

Example: [$H$ $x$ $y$ $e$] [$h$ $Dx$ $Dy$ $E$]
         [0 1 1 1 4 1 1 1 3]
        (  e y x H   E Dy Dx h )
         [[1 1 1 1   1 1  1  0]
          [1 0 0 0   0 0  0  0]
          [0 0 0 0   0 1  0  0]
           ........

cf. polynomial_ring, ring_of_..., groebner.
```

69. setcontext

```
<< class.Context setcontext >>
Change the CurrentContextp (and the user dictionary) to class.Context.
Ex. (mynew) StandardContextp newcontext /cp set cp setcontext
cf. system_variable(CurrentContextp), newcontext,
    show_user_dictionary, StandardContextp.
```

70. setstack

```
<< class.OperandStack setstack >>
Change the current operand stack to class.OperandStack.
Ex. 1000 newstack /st set  st setstack
cf. newstack, stdstack
```

71. show_ring

```
<< show_ring >>
cf. set_up_ring.
```

72. show_systemdictionary

```
<< show_systemdictionary >>  << show_user_dictionary >>
```

73. show_user_dictionary

```
<< show_systemdictionary >>  << show_user_dictionary >>
```

74. spol

```
<< f  g spol  [ [c d] r] >>
poly f, g, c, d, r;
r is the s-polynomial of f and g.
 r = c f + d g
```

75. stdstack

```
<< stdstack >>
Change the current operand stack to the standard stack.
cf. newstack, setstack
```

76. sub

```
obj1 obj2 sub obj3
obj3 = obj1 - obj2.
```

77. supercontext

```
<< class.Context supercontext class.Context2 >>
cf. setcontext, newcontext
```

78. supmsg2

```
<< obj2 obj3 { executable array } supmsg2 >>
Context is the super of obj2 or obj3.
cf. sendmsg, sendmsg2, primmsg.
```

79. switch_function

```
<< (function) (name) switch_function >>
<< (report) (function) switch_function value >>
cf. print_switch_status, getOptions, restoreOptions.
```

80. system

```
<< command system >>
string command;
Execute unix command.
Example: (ls) system
```

81. system_variable

```
<< [(key word)] system_variable result >> or
<< [(key word) value] system_variable result >>
<< [(key word) optional arg] system_variable result >>
The primitive returns the value of a given system constant.
The primitive sets the value of a given system constant.
PrintDollar [0,1,2], Wrap, P, N, NN, M, MM, L, LL, C, CC,ringName,
CurrentRingp, Verbose, UseCriterion1, UseCriterion2B,
ReduceLowerTerms, CheckHomogenization, Homogenize, Sugar, Homogenize_vec,
Statistics, StackPointer, StandardOperandStack,
ErrorStack, ErrorMessageMode, WarningMessageMode,
CatchCtrlC, Strict, CurrentContextp, NullContextp, Strict2,
SigIgn, KSPushEnvMode, PrimitiveContextp, KanGBmessage,
orderMatrix, gbListTower, Schreyer,
outputOrder, multSymbol, variableNames, Version,
AvoidTheSameRing, RingStack, LeftBracket, RightBracket,
AutoReduce, SecureMode
Example: [(N)] system_variable
                             (maximum number of variables)/2.
         [(x) (var) 3] system_variable
                              The 3rd variable.
         [(D) (var) 4] system_variable
                              The 4th differential operator.
         [(PrintDollar) 0] system_variable
                          Set the global variable PrintDollar to 0
```

82. test

```
The keyword <<test>> cannot be found.
Type in ? in order to see system dictionary.
```

83. to_records

```
<< ({arg1,arg2,...,argn})  to_records (arg1) ... (argn) n >>
Example: ({x,y2}) to_records ===> (x) (y2) 2
Example: (x,y2)   to_records ===> (x) (y2) 2
Example: [(x,y2) to_records pop] ===> [(x) (y2)]
```

84. writestring

```
<< fd s writestring >>
file fd; string s;
Write the string s to the file fd.
cf. [(PrintDollar)] system_variable, file, closefile
```

2 Macro Operators

1.

```
gate keeper
```

2. `,,`

```
 string ring ,, polynomial
Parse the <<string>> as an element in the <<ring>> and returns
the polynomial.
cf. define_ring, define_qring, ring_def
Example: [(x,y) ring_of_polynomials [[(x) 1]] weight_vector 7]define_ring
        /myring set
        ((x+y)^4) myring ,, /f set
```

3. `-intInfinity`

```
-intInfinity = -999999999
```

4. `.`

```
string . polynomial
Parse the string as a polynomial in the current ring and put it on the stack
Example ( (x+2)^3 ) .
```

5. `..`

```
 string .. universalNumber
Parse the << string >> as a universalNumber.
Example:  (123431232123123).. /n set
```

6. `..int`

```
universalNumber ..int int
```

7. `::`

```
Pop the top of the stack and print it.
```

8. `;`

```
Output the prompt sm1>
```

9. `==`

```
obj ==
Print obj
```

10. `@@@.Dsymbol`

```
@@@. is the prefix for the global control variables and global functions.
@@@.quiet   :  1 ---> quiet mode,  0 ---> not. cf. sm1 -q
@@@.esymbol : (e_) is the standard value. It is used to express
              vectors internally. cf. fromVectors, toVectors
@@@.Dsymbol :  (D) is the standard value for differential operator.
                cf. ring_of_differential_operators
@@@.diffEsymbol :  (E) is the standard value for difference operator.
                cf. ring_of_difference_operators
@@@.Qsymbol :  (Q) is the standard value for q-difference operator.
                cf. ring_of_qdifference_operators
```

11. `@@@.Qsymbol`

```
@@@. is the prefix for the global control variables and global functions.
@@@.quiet   :  1 ---> quiet mode,  0 ---> not. cf. sm1 -q
@@@.esymbol : (e_) is the standard value. It is used to express
              vectors internally. cf. fromVectors, toVectors
@@@.Dsymbol :  (D) is the standard value for differential operator.
                cf. ring_of_differential_operators
@@@.diffEsymbol :  (E) is the standard value for difference operator.
                cf. ring_of_difference_operators
@@@.Qsymbol :  (Q) is the standard value for q-difference operator.
                cf. ring_of_qdifference_operators
```

12. `@@@.diffEsymbol`

```
@@@. is the prefix for the global control variables and global functions.
@@@.quiet   :  1 ---> quiet mode,  0 ---> not. cf. sm1 -q
@@@.esymbol : (e_) is the standard value. It is used to express
              vectors internally. cf. fromVectors, toVectors
@@@.Dsymbol :  (D) is the standard value for differential operator.
                cf. ring_of_differential_operators
@@@.diffEsymbol :  (E) is the standard value for difference operator.
                cf. ring_of_difference_operators
@@@.Qsymbol :  (Q) is the standard value for q-difference operator.
                cf. ring_of_qdifference_operators
```

13. `@@@.esymbol`

```
@@@. is the prefix for the global control variables and global functions.
@@@.quiet   :  1 ---> quiet mode,  0 ---> not. cf. sm1 -q
@@@.esymbol : (e_) is the standard value. It is used to express
              vectors internally. cf. fromVectors, toVectors
@@@.Dsymbol :  (D) is the standard value for differential operator.
                cf. ring_of_differential_operators
@@@.diffEsymbol :  (E) is the standard value for difference operator.
                cf. ring_of_difference_operators
@@@.Qsymbol :  (Q) is the standard value for q-difference operator.
                cf. ring_of_qdifference_operators
```

14. `@@@.quiet`

```
@@@. is the prefix for the global control variables and global functions.
@@@.quiet   :  1 ---> quiet mode,  0 ---> not. cf. sm1 -q
@@@.esymbol : (e_) is the standard value. It is used to express
              vectors internally. cf. fromVectors, toVectors
@@@.Dsymbol :  (D) is the standard value for differential operator.
                cf. ring_of_differential_operators
@@@.diffEsymbol :  (E) is the standard value for difference operator.
                cf. ring_of_difference_operators
@@@.Qsymbol :  (Q) is the standard value for q-difference operator.
                cf. ring_of_qdifference_operators
```

15. `ArrayP`

```
IntegerP, LiteralP, StringP, ExecutableArrayP, ArrayP, PolyP, FileP, RingP,
UniversalNumberP, RationalFunctionP, ClassP, DoubleP
      return data type identifiers.
Example:  7 tag IntegerP eq  ---> 1
```

16. `ClassP`

```
IntegerP, LiteralP, StringP, ExecutableArrayP, ArrayP, PolyP, FileP, RingP,
UniversalNumberP, RationalFunctionP, ClassP, DoubleP
      return data type identifiers.
Example:  7 tag IntegerP eq  ---> 1
```

17. `DoubleP`

```
IntegerP, LiteralP, StringP, ExecutableArrayP, ArrayP, PolyP, FileP, RingP,
UniversalNumberP, RationalFunctionP, ClassP, DoubleP
      return data type identifiers.
Example:  7 tag IntegerP eq  ---> 1
```

18. ExecutableArrayP

```
IntegerP, LiteralP, StringP, ExecutableArrayP, ArrayP, PolyP, FileP, RingP,
UniversalNumberP, RationalFunctionP, ClassP, DoubleP
      return data type identifiers.
Example:  7 tag IntegerP eq  ---> 1
```

19. IntegerP

```
IntegerP, LiteralP, StringP, ExecutableArrayP, ArrayP, PolyP, FileP, RingP,
UniversalNumberP, RationalFunctionP, ClassP, DoubleP
      return data type identifiers.
Example:  7 tag IntegerP eq  ---> 1
```

20. Libraries

```
Doc/appell.sm1   : It generates Appell differential equations.
Doc/bfunction.sm1: Computing the b-function of a given polynomial
                   by using Groebner basis. Written by T.Oaku.
Doc/factor-a.sm1: A sample interface to factor polynomials by risa/asir.
Doc/gkz.sm1     : It generates GKZ systems for given A and b.
Doc/hol.sm1     : Basic package for holonomic systems. Holonomic rank,
                  characteristic ideal, singular locus.
Doc/resol0.sm1  : Constructing Schreyer resolutions. tower.sm1, tower-sugar.sm1
Doc/rest0.sm1   : Computing the restriction (inverse image) as a complex.
                  This package is under developement by T.Oaku.
                  cf. restall_s.sm1, resol0.sm1
```

21. LiteralP

```
IntegerP, LiteralP, StringP, ExecutableArrayP, ArrayP, PolyP, FileP, RingP,
UniversalNumberP, RationalFunctionP, ClassP, DoubleP
      return data type identifiers.
Example:  7 tag IntegerP eq  ---> 1
```

22. Loadall

```
Loadall loads the packages  bfunction.sm1, hol.sm1, gkz.sm1, appell.sm1,
resol0.sm1
```

23. PolyP

```
IntegerP, LiteralP, StringP, ExecutableArrayP, ArrayP, PolyP, FileP, RingP,
UniversalNumberP, RationalFunctionP, ClassP, DoubleP
      return data type identifiers.
Example:  7 tag IntegerP eq  ---> 1
```

24. `RationalFunctionP`

```
IntegerP, LiteralP, StringP, ExecutableArrayP, ArrayP, PolyP, FileP, RingP,
UniversalNumberP, RationalFunctionP, ClassP, DoubleP
      return data type identifiers.
Example:  7 tag IntegerP eq  ---> 1
```

25. `RingP`

```
IntegerP, LiteralP, StringP, ExecutableArrayP, ArrayP, PolyP, FileP, RingP,
UniversalNumberP, RationalFunctionP, ClassP, DoubleP
      return data type identifiers.
Example:  7 tag IntegerP eq  ---> 1
```

26. `SmallRing`

```
SmallRing is the ring of polynomials Q[t,x,T,h].
```

27. `StringP`

```
IntegerP, LiteralP, StringP, ExecutableArrayP, ArrayP, PolyP, FileP, RingP,
UniversalNumberP, RationalFunctionP, ClassP, DoubleP
      return data type identifiers.
Example:  7 tag IntegerP eq  ---> 1
```

28. `UniversalNumberP`

```
IntegerP, LiteralP, StringP, ExecutableArrayP, ArrayP, PolyP, FileP, RingP,
UniversalNumberP, RationalFunctionP, ClassP, DoubleP
      return data type identifiers.
Example:  7 tag IntegerP eq  ---> 1
```

29. `Version`

```
Packages must be loaded in a proper version of kan/sm1.
It can be checked as follows.
Example:
   /factor-a.version (2.981101) def
   factor-a.version [(Version)] system_variable gt
   { (This package requires the latest version of kan/sm1) message
     (Please get it from http://www.math.kobe-u.ac.jp/KAN) message
     error
   } { } ifelse
```

30. action

```
 f g action p
<poly> f,g,p
Act f on g. The result is p. The homogenization variable h is put to 1.
We can use diff0 only in the ring of differential operators.
Example: [(x) ring_of_differential_operators 0] define_ring
         (Dx^2). (x^2). action ::
```

31. adjoint

```
f xlist adjoint g
poly f; string xlist; poly g;
g is the adjoint operator of f.
The variables to take adjoint are specified by xlist.
Example: [(x,y) ring_of_differential_operators 0] define_ring
          (x^2 Dx - y x Dx Dy-2). (x,y) adjoint
          ((-Dx) x^2 - (-Dx) (-Dy) x y -2). dehomogenize sub :: ==> 0
```

32. and

```
obj1 obj2 and bool
```

33. append

```
list1 obj2 append list3
Ex.  [1 2]  3 append  [1 2 3]
```

34. break

```
bool break
```

35. bugs

```
All known bugs are fixed.
```

36. cancelCoeff

```
f cancelcoeff g
poly f,g;
Factor out the gcd of the coefficients.
Example: (6 x^2 - 10 x). cancelCoeff
See also gbext.
```

37. carN

```
[f1 ... fm]  n carN  [f1 ... fn]
carN extracts the first n elements from the list.
```

38. cat

```
a cat s
array a ; string s;
cat converts each entry of << a >> to a string and concatenates them.
Example: [ (x) 1 2] cat ==> (x12)
```

39. complement

```
 set universal_set complement complement_set
Example: [1 2] [3 4 1 2 5] complement ::
```

40. ctrlC-hook

```
When ctrl-C is pressed, this function is executed.
User can define one's own ctrlC-hook function.
```

41. dc

```
Abbreviation of data_conversion.
```

42. define_qring

```
 [varlist ring_of_q_difference_operators order characteristic] define_qring
    Pointer to the ring.
Example: [$x,y$ ring_of_q_difference_operators $Qx,Qy$ elimination_order
          0] define_qring
cf. define_ring, set_up_ring@ <coefficient ring>, ring_def, << ,, >>
```

43. define_ring

```
 [varlist ring_of_??? order characteristic options] define_ring
 Pointer to the ring.
Example: [(x,y,z) ring_of_polynomials [[(x) 100 (y) 10 (z) 1]] weight_vector
          11] define_qring
Example: [(x,y) ring_of_polynomials [[(x) 1]] weight_vector 0 ] define_ring
         /R set
```

```
cf. define_qring, set_up_ring@ <coefficient ring>
    <<ring_of_???>> ring_of_polynomials, ring_of_differential_operators,
    ring_of_difference_operators
    <<order>> elimination_order, weight_vector
    ring_def,ring , << ,, >>
```

44. dehomogenize

```
obj dehomogenize obj2
dehomogenize puts the homogenization variable to 1.
Example:  (x*h+h^2). dehomogenize ::    x+1
```

45. diff0

```
f v n diff0 fn
<poly> fn, v ; <integer>  n ; <poly> fn
fn = v^n f where v^n is the operator to take the n-th differential.
We can use diff0 only in the ring of differential operators.
Example: [(x) ring_of_differential_operators 0] define_ring
         (x^10-x). (Dx). 1 diff0 ::
```

46. distraction

```
f [ list of x-variables ] [ list of D-variables ] [ list of s-variables ]
   distraction  result
Example: (x Dx Dy + Dy). [(x). (y).] [(Dx). (Dy).] [(x). (y).] distraction
```

47. distraction2

```
f [ list of x-variables ] [ list of D-variables ] [ list of s-variables ]
   distraction2  result
Example 1: [(x,y) ring_of_differential_operators 0] define_ring
  (x^2 Dx Dy + x Dy). [(x). (y).] [(Dx). (Dy).] [(x). (y).] distraction2
Example 2: (x^4 Dx^2 + x^2). [(x).] [(Dx). ] [(x).] distraction2
```

48. eliminatev

```
[g1 g2 g3 ...gm] [list of variables] eliminatev [r1 ... rp]
Example: [(x y z - 1). (z-1). (y-1).] [(x) (y)] eliminatev [ z-1 ]
```

49. etag

```
obj etag integer
etag returns extended object tag. cf. kclass.c
```

50. **evecw**

```
size position weight evecw  [0 0 ... 0 weight 0 ... 0]
Example: 3 0 113 evecw ===> [113  0  0]
```

51. **evenQ**

```
number evenQ bool
```

52. **factor**

```
Load Doc/factor-a.sm1, then factorization can be computed by invoking
asir, of which ftp cite is at http://www.math.kobe-u.ac.jp
```

53. **factorial**

```
f n factorial g
integer n,  g is f (f-1) ... (f-n+1)
```

54. **findIntegralRoots**

```
 f findIntegralRoots vlist
poly f; list of integers vlist;
string f; list of integers vlist;
f is a polynomials in one variable s. vlist the list of integral roots sorted.
Example: (s^4-1) findIntegralRoots
```

55. **flatten**

```
list flatten list2
Flatten the list.
Example 1: [ [1 2 3] 4 [2]] flatten    ===> [1 2 3 4 2]
```

56. **fromVectors**

```
[v1 v2 ...] fromVectors [s1 s2 ...]
array of poly : v1, v2, ... ; poly : s1, s2 ....
cf. toVectors. <<e_>> varaible is assumed to be the last
    variable in x.  @@@.esymbol
Example: [(x,y) ring_of_differential_operators 0] define_ring
 [(x).  (y).] /ff set
 [ff ff] fromVectors ::
```

57. from_records

```
[s1 s2 s3 ... sn] from_records (s1,s2,...,sn,)
Example : [(x) (y)] from_records ::    (x,y,)
cf. to_records
```

58. ge

```
obj1 obj2 ge bool
greater than or equal
```

59. gensym

```
x i gensym xi
string x; integer i; string xi
It generate a string x indexed with the number i.
Example:  (Dx) 12 gensym (Dx12)
```

60. getOptions

```
 getOptions [<options for system_variable>  <options for switch_function>]
cf. restoreOptions
```

61. getOrderMatrix

```
obj getOrderMatrix m
array  m
getOrderMatrix obtains the order matrix from obj.
If obj is a polynomial, it returns the order matrix associated to
the polynomial.
If obj is an array, it returns an order matrix of an element.
```

62. getRing

```
obj getRing rr
ring rr;
getRing obtains the ring structure from obj.
If obj is a polynomial, it returns the ring structure associated to
the polynomial.
If obj is an array, it recursively looks for the ring structure.
```

63. getVariableNames

```
getVariableNames list-of-variables
Example: getVariableNames :: [e,x,y,E,H,Dx,Dy,h]
```

64. getvNames

```
getvNames vlist
list vlist
It returns of the list of the variables in the order x0, x1, ..., D0, ...
Use with [(variableNames) vlist] system_variable.
cf. nlist getvNames0 vlist is used internally. cf. getvNamesC
```

65. getvNamesC

```
getvNamesC vlist
list vlist
It returns of the list of the variables in the order 0, 1, 2, ...
(cmo-order and output_order).
cf. getvNames
```

66. getvNamesCR

```
obj getvNamesCR vlist
obj ring | poly ; list vlist
It returns of the list of the variables in the order 0, 1, 2, ... (cmo-order)
for <<obj>>.
Example: ( (x-2)^3 ). /ff set
         [(x) ring_of_differential_operators 0] define_ring ff getvNamesCR ::
```

67. groebner_sugar

```
groebner_sugar computes Groebner basis by the sugar strategy.
Format of arguments and results are as same as that of groebner.
See groebner.  When you compute in the ring of polynomials,
it is recommended to turn on the switch [(UseCriterion1) 1] system_variable
Note that groebner_sugar does not use the homogenized Weyl algebra.
So, it does work only for term orders. Never use negative weight vectors.
```

68. hilb

```
base vlist hilb f
array of poly base; array of poly vlist;  poly f;
array of string base; array of string vlist; poly f;
array of string base; string vlist; number m; poly f;
f is the hilbert function (a_d x^d + ...)/m!
The << base >> should be a reduced Grobner basis.
Or, when the << base >> is an array of string,
```

```
all entries should be monomials.
Example: [(x^2) (x y )] (x,y)  hilb ::   h + 2
Example: [(x^2) (y^2)] (x,y) hilb  4
Example: [(x^2) (y^2) (x y)] [(x) (y)] hilb ::  3
cf. hilbert,   hilbReduce
```

69. hilbReduce

```
[f,g] v hilbReduce p
output of hilbert [f,g];  string v; poly p
p is  (g/(f!))*deg(g)!
 [(x) (y^3)] (x,y,z) hilbert (h) hilbReduce
```

70. hilbert

```
base vlist hilbert [m f]
array of poly base; array of poly vlist; number m; poly f;
array of string base; array of string vlist; number m; poly f;
array of string base; string vlist; number m; poly f;
[m f] represents the hilbert function (a_d x^d + ...)/m! where f=a_d x^d + ...
The << base >> should be a reduced Grobner basis.
Or, when the << base >> is an array of string,
all entries should be monomials.
Example: [(x^2) (x y )] (x,y)  hilbert ::  [2, 2 h + 4]
Example: [(x^2) (y^2)] (x,y) hilbert (h) hilbReduce ::  4
Example: [(x^2) (y^2) (x y)] [(x) (y)] hilbert (h) hilbReduce ::  3
cf. hilb,   hilbReduce
```

71. intInfinity

```
intInfinity = 999999999
```

72. ip1

```
 [v1 ... vn] [w1 ... wn] m ip1 [f1 ... fs]
<poly> v1 ... vn ; <integer> w1 ... wn m
<poly> f1 ... fs
Example: [(x,y) ring_of_differential_operators 0] define_ring
         [(Dx). (Dy).] [2 1] 3 ip1  ::    [(2 Dx Dy). (Dy^3).]
         Returns Dx^p Dy^q such that  2 p + 1 q = 3.
```

73. isArray

```
obj isArray bool
```

74. `isClass`

```
obj isClass bool
```

75. `isDouble`

```
obj isDouble bool
```

76. `isInteger`

```
obj isInteger bool
```

77. `isPolynomial`

```
obj isPolynomial bool
```

78. `isRational`

```
obj isRational bool
```

79. `isString`

```
obj isString bool
```

80. `isUniversalNumber`

```
obj isUniversalNumber bool
```

81. `join`

```
list1 list2 join list3
Ex.  [1 2] [3 [4 (ab)]] join  [1 2 3 [4 (ab)]]
```

82. `laplace0`

```
f [v1 ... vn] laplace0 g
poly f ; string v1 ... vn ; poly g;
array of poly f ; string v1 ... vn ; array of poly g;
 g is the lapalce transform of f with respect to variables v1, ..., vn.
Example: (x Dx + y Dy + z Dz). [(x) (y) (Dx) (Dy)] laplace0
 x --> -Dx, Dx --> x,  y --> -Dy, Dy --> y.
```

83. `le`

```
obj1 obj2 le bool
less than or equal
```

84. `makeRingMap`

```
 rule  ring1 ring2 makeRingMap maptable
makeRingMap is an auxiliary function for the macro ringmap. See ringmap
```

85. `maxInArray`

```
 [v1 v2 ....] maxInArray m
m is the maximum in [v1 v2 ...].
The position of m is stored in the global variable maxInArray.pos.
```

86. `memberQ`

```
element array(list) memberQ bool
```

87. `message`

```
string message
Output the string to the screen with the newline.
cf. messagen.
Example:  (Hello world) message
```

88. `message-quiet`

```
s message-quiet
string s;
It outputs the message s when @@@.quiet is not equal to 1.
@@@.quiet is set to 1 when you start sm1 with the option -q.
```

89. `messagen`

```
string messagen
Output the string to the screen without the newline.
cf. message.
Example:  (Hello world) messagen
```

90. messagen-quiet

```
s messagen-quiet
string s;
It outputs the message s without the newline when @@@.quiet is not equal to 1.
@@@.quiet is set to 1 when you start sm1 with the option -q.
```

91. ngcd

```
nlist ngcd d
list of numbers nlist; number d;
d is the gcd of the numbers in nlist.
Example: [(12345).. (67890).. (98765)..] ngcd
```

92. nl

```
nl is the newline character.
Example: [(You can break line) nl (here.)] cat message
```

93. not

```
int not int
1 not 0
0 not 1
```

94. npower

```
obj1 obj2 npower obj3
npower returns obj1^obj2 as obj3
The difference between power and npower occurs when we compute f^0
where f is a polynomial.
power returns number(universalNumber) 1, but npower returns 1
in the current ring.
```

95. null

```
<<null>> returns null object
```

96. or

```
obj1 obj2 or bool
```

97. ord_w

```
ff [v1 w1 v2 w2 ... vm wm] ord_w d
poly ff; string v1; integer w1; ...
order of ff by the weight vector [w1 w2 ...]
Example: [(x,y) ring_of_polynomials 0] define_ring
         (x^2 y^3-x). [(x) 2 (y) 1] ord_w ::
```

98. output

```
obj output
Output the object to the standard file sm1out.txt
```

99. output_order

```
 [(v1) (v2) ...] output_order
Set the order of variables to print for the current ring.
cf. system_variable
Example:  [(y) (x)] output_order
          (x*y). ::   ===> y*x
```

100. oxasir

```
ox_asir is an asir server which is complient to open xxx protocol.
This module is necessary for efficient execution factorization
of b-functions in annfs, deRham.  cf. lib/oxasir.sm1
ox_asir has not yet been put on the ftp cite. 12/17, 1998.
```

101. plugin-cmo

```
Cmo is for supporting Open XM communication protocol.
See http://www.math.kobe-u.ac.jp/openxxx for the latest info.
This document is incomplete. See lib/ox.sm1.
cmoObjectToCmo obj, cmoDumpCmo cmo-obj, cmoCmoToObject cmo-obj,
cmoCmoToStream cmo-obj stream, cmoStreamToCmo stream, cmoToStream stream,
cmoFromStrem stream, cmoMathCap,
cmoDebugCMO inteter,  cf. plugin-ox oxPrintMessage
cmoOxSystem, cmoOxSystemVersion, cmoLispLike
```

102. plugin-file2

```
[(fp2fdopen) fd-num] extension file-fp2
[(fp2fflush) file-fp2] extension result
[(fp2fclose) file-fp2] extension result
[(fp2fgetc) file-fp2] extension c
[(fp2fputc) c file-fp2] extension c
[(fp2dumpBuffer) file-fp2] extension c
[(fp2openForRead) name] extension fd-num
[(fp2openForWrite) name] extension fd-num
[(fp2clearReadBuf) file-fp2] extension c
```

103. plugin-ox

```
OX is for supporting Open XM communication protocol.
See http://www.math.kobe-u.ac.jp/openxxx for the latest info.
This document is incomplete. See lib/ox.sm1 and plugin-cmo.
oxCreateClient, oxCreateClientFile, oxReq, oxGet, oxGetFromControl,
oxMultiSelect, oxWatch, oxCloseClient, oxSerial, oxGenPortFile,
oxRemovePortFile, oxGenPass, oxGetPort, oxCreateClient2,
oxPrintMessage cf. plugin-cmo, cmoDebugCMO
```

104. plugin-sm1.socket

```
[    [(sm1.socket) key [args] ] extension result-object ]
[     key  :  open, connect, accept, select, mselect, read, write, readByte, writeByte, close. , Note that read and write are used for only string data. ]
[    See also Kan/debug/server.sm1 ]
```

105. pmat

```
f pmat
array f;
f is pretty printed.
```

106. popEnv

```
envlist popEnv
cf. pushEnv
```

107. position

```
set element position number
Example: [(cat) (dog) (hot chocolate)] (cat) position ===> 0.
```

108. power

```
obj1 obj2 power obj3
obj3 is (obj1)^(obj2). cf. npower
Example:  (2). 8 power ::  ===>  256
```

109. pushEnv

```
keylist pushEnv envlist
array of string keylist, array of [string object] envlist;
Values <<envlist>> of the global system variables specified
by the <<keylist>> is push on the stack.
keylist is an array of keywords for system_variable.
cf. system_variable, popEnv
Example:  [(CurrentRingp) (KanGBmessage)] pushEnv
```

110. pushVariables

```
varlist pushVariables pushed-variables
The macro is used to define local variables in a macro.
Example : /foo {
           /arg1 set
          [/abc /cd] pushVariables
          [ /abc arg1 def
            /cd abc 1 add def
            /arg1 cd def
          ] pop
          popVariables
          arg1
          } def
  10 foo ::    returns 11
    In the example, abc and cd are local variables.
```

111. putUsages

```
[(key word) [(explanation line 1) (explanation line 2) ... ]] putUsages
```

112. reducedBase

```
base reducedBase reducedBase
<<reducedBase>> prunes redundant elements in the Grobner basis <<base>> and
returns <<reducedBase>>.
Ex. [(x^2+1). (x+1). (x^3).] reducedBase ---> [(x+1).]
```

113. reduction-noH

```
f g reduction-noH r
poly f; array g; array r;
Apply the normal form algorithm for f with the set g. All computations are
done with the rule Dx x = x Dx +1, i.e., no homogenization, but other
specifications are the same with reduction. cf. reduction
g should be dehomogenized.
```

114. rest

```
array rest the-rest-of-the-array
Ex. [1 2 [3 0]] rest ===> [2 [3 0]]
```

115. restoreOptions

```
[<options for system_variable>  <options for switch_function>] restoreOptions
cf. getOptions
```

116. reverse

```
array reverse reversed_array
```

117. ring_def

```
ring ring_def
Set the current ring to the <<ring>>
Example: [(x,y) ring_of_polynomials [[(x) 1]] weight_vector 0 ] define_ring
          /R set
          R ring_def
In order to get the ring object R to which a given polynomial f belongs,
one may use the command
          f (ring) data_conversion /R set
cf. define_ring, define_qring, system_variable, poly (ring) data_conversion
cf. << ,, >>
```

118. ring_of_difference_operators

```
string ring_of_difference_operators

This command has not been maintained since 1996. So, there may be
troubles if it is used with new features of kan/sm1.

This command is used to define a ring of difference operators
with the macros define_ring, weight_vector and elimination_order.
The user cannot use the variable names h, e, H and E.
cf. show_ring, system_variable --- CurrentRing
    print_switch_status, switch_function
```

```
Example: [(x,y,z) ring_of_difference_operators (Ex,Ey) elimination_order 0] define_ring
         Ex,Ey, and Ez are corresponding difference operators to the
         space variables x, y and z ( Ex x = (x+1) Ex).
```

119. ring_of_differential_operators

```
string ring_of_differential_operators
This command is used to define a ring of differential operators
with the macros define_ring, weight_vector and elimination_order.
The user cannot use the variable names h, e, H and E.
cf. show_ring, system_variable --- CurrentRing
    print_switch_status, switch_function
Example: [(x,y,z) ring_of_differential_operators (Dx,Dy) elimination_order 0] define_ring
         Dx,Dy, and Dz are corresponding differential operators to the
         space variables x, y and z.
```

120. ring_of_polynomials

```
string ring_of_polynomials
This command is used to define a ring of polynomials with the macros
define_ring, weight_vector and elimination_order.
The user cannot use the variable names h, e, H and E.
cf. show_ring, system_variable --- CurrentRing
    print_switch_status, switch_function
Example: [(x,y,z) ring_of_polynomials (x,y) elimination_order 0] define_ring
```

121. ring_of_q_difference_operators

```
string ring_of_q_difference_operators

This command has not been maintained since 1996. So, there may be
troubles if it is used with new features of kan/sm1.

This command is used to define a ring of q-difference operators
with the macros define_qring, weight_vector and elimination_order.
Note that you should use this macro with define_qring instead of
define_ring.
The user cannot use the variable names h, e, q and E.
cf. show_ring, system_variable --- CurrentRing
    print_switch_status, switch_function
Example: [(x,y,z) ring_of_q_difference_operators (Qx,Qy) elimination_order 0] define_qring
         Qx,Qy, and Qz are corresponding q-difference operators to the
         space variables x, y and z ( Qx x = (q) x Q).
```

122. ringmap

```
f mapTable ringmap r
f is mapped to r where the map is defined by the mapTable, which is generated
```

```
by makeRingMap as follows:
 rule  ring1 ring2 makeRingMap maptable
Example:
[(x,y) ring_of_differential_operators ( ) elimination_order 0] define_ring
/R1 set
[(t,y,z) ring_of_differential_operators ( ) elimination_order 0] define_ring
/R2 set
[[(x) (Dx)] [((t-1) Dt) (z)]] /r0 set
r0 R1 R2 makeRingMap /maptable set
(Dx-1) R1 ,, /ff set
 ff maptable ringmap ::
```

123. shell

```
[gate-keeper f1 f2 ... fm] shell result
Sort the list. Gate-keeper should be the smallest element
```

124. swap01

```
[ .... ] swap01 [....]
Examples: [(x,y) ring_of_polynomials (x) elmination_order 0] swap01
          define_ring
```

125. swap0k

```
[ .... ] k swap0k [....]
Examples: [(x,y) ring_of_polynomials (x) elmination_order 0] 1 swap0k
          define_ring
swap01 == 1 swap0k
```

126. tag

```
obj tag integer
tag returns datatype.
cf. data_conversion
Example: 2 tag IntegerP eq ---> 1
```

127. timer

```
 { codes } timer
It outputs the execution time to execute << codes >>.
If you type in ctrl-C while you are executing this macro,
you need to type in set_timer after you type in ctrl-C to reset
the timer.
```

128. toString

```
obj toString
Convert obj to a string.
Example: [ 1 (x+1). [ 2 (Hello)]] toString ==> $[ 1 , x+1 , [ 2 , Hello ]  ]$
```

129. toVectors

```
obj toVectors vec
Convert the internal expression of vector into the array <<vec>>.
obj can be  [n [g1 ... gm]] where n is the length of the vector.
Example 1: [(x) ring_of_polynomials 0] define_ring
           (e_ + 2).  toVectors ::
        --->  [2  , 1]
Example 2: [3 (e_ +2).] toVectors ::
        ---> [2 , 1 , 0 ]
Example 3: [(e_+2). (e_^2+1).] toVectors ::
        ---> [[2 , 1] [1 , 0 , 1]]
Example 4: [2 [(e x + 1). (x+1).]] toVectors
           ===> [ [1 , x] [x+1 , 0] ]
```

130. toVectors2

```
Remained for the compatibility. cf. toVectors
```

131. tolower

```
string tolower string2
Capital letters in string are converted to lower case letters.
Example:  (Hello World) tolower :: (hello world)
```

132. transpose

```
 matrix transpose transposed_matrix
```

133. weight_vector

```
[x-list d-list params] [[(name) weight ...] [...] ...] weight_vector
[x-list d-list params order]
Example:
   [(x,y,z) ring_of_polynomials [[(x) 100 (y) 10]] weight_vector 0]
   define_ring
```

134. `weightv`

```
array weightv weight_vector_for_init
cf. init
Example: /w [(x) 10 (h) 2] weightv def
        ((x-h)^10).  w init ::
```

3 Macros in libraries

The macros described in this section can be used by loading library files by the command (cohom.sm1) run `ox_asir` functions require open asir module.
var.sm1 : Version 3/7, 1997
complex.sm1 : 1999, 9/28, res-div, res-solv, res-kernel-image, res-dual In this package, complex is expressed in terms of matrices.
getPathNameSm1: The file bin/ox could not be found neigher in LOAD_SM1_PATH=../../../lib/sm1 nor in OpenXM_HOME/lib/sm1=/home/taka/OpenXM/lib/sm1 nor in /usr/local/lib/sm1
getPathNameSm1: The file bin/ox could not be found neigher in LOAD_SM1_PATH=../../../lib/sm1 nor in OpenXM_HOME/lib/sm1=/home/taka/OpenXM/lib/sm1 nor in /usr/local/lib/sm1
lib/intw.sm1, Version 1999, 6/13. Package for integration with a generic weight.
Warning: This package requires cohom.sm1
Warning: This package requires oxasir.sm1 integral-k1 wbf intwbf tensor0 wTensor0 intwbfRoots wbfRoots wIntegration0 wRestriction0 ann-t-f bf-111 wdeRham0

1. `Offverbose`

```
The keyword <<Offverbose>> cannot be found.
Type in ? in order to see system dictionary.
```

2. `Onverbose`

```
The keyword <<Onverbose>> cannot be found.
Type in ? in order to see system dictionary.
```

3. `ann-t-f`

```
ann-t-f returns the annihilating ideal of delta(t-f(x))
Example: [(x^3-y^2) [(t) (x) (y)]] ann-t-f
```

4. `annfs`

```
 [ f v m r0] annfs g
It returns the annihilating ideal of f^m where r0 must be smaller
or equal to the minimal integral root of the b-function.
Or, it returns the annihilating ideal of f^r0, r0 and the b-function
where r0 is the minial integral root of b.
For the algorithm, see J. Pure and Applied Algebra 117&118(1997), 495--518.
Example 1: [(x^2+y^2+z^2+t^2) (x,y,z,t) -1 -2] annfs ::
           It returns the annihilating ideal of (x^2+y^2+z^2+t^2)^(-1).
Example 2: [(x^2+y^2+z^2+t^2) (x,y,z,t)] annfs ::
           It returns the annihilating ideal of f^r0 and [r0, b-function]
           where r0 is the minimal integral root of the b-function.
Example 3: [(x^2+y^2+z^2) (x,y,z) -1 -1] annfs ::
Example 4: [(x^3+y^3+z^3) (x,y,z)] annfs ::
Example 5: [((x1+x2+x3)(x1 x2 + x2 x3 + x1 x3) - t x1 x2 x3 )
            (t,x1,x2,x3) -1 -2] annfs ::
           Note that the example 4 uses huge memory space.
```

5. `appell1`

```
param appell1 c
array param; array c;
appell1 returns the Lauricella function F_D(a,b_1, ..., b_n,c; x_1,...,x_n)
for the parameter << param >> = [a, c, b_1, ..., b_n].
In case of n=2, the function is called the Appell function F_1.
c = [ generators,  variables ]
Note that for a special set of parameters, the returned differential equation
is not holonomic, e.g., [[1 2 3 4]] appell1 rank ::
This happens because we do not included the Euler-Darboux operators
in the return value of appell1. It will be included in a future.
Example: [ [1 -4 -2 5 6] ] appell1 rank ::
For details, see P.Appell et Kampe de Feriet, Fonction hypergeometrique
et hyperspheriques -- polynomes d'Hermite, Gauthier-Villars, 1926.
```

6. `appell4`

```
param appell4 c
array param; array c;
appell4 returns the Lauricella function F_C(a,b, c_1, ..., c_n; x_1,...,x_n)
for the parameter << param >> = [a, c, b_1, ..., b_n].
In case of n=2, the function is called the Appell function F_4.
c = [ generators,  variables ]
Note that for a special set of parameters, the returned differential equation
is not holonomic, e.g., [[1 2 3 4]] appell4 rank ::
Example: [ [1 -4 -2 5 6] ] appell4 rank ::
```

7. `asir`

```
The keyword <<asir>> cannot be found.
Type in ? in order to see system dictionary.
```

8. asirconnectr

```
The keyword <<asirconnectr>> cannot be found.
Type in ? in order to see system dictionary.
```

9. bf-111

```
 [ideal vlist rest-vlist bf-111] bf-111
Compute the b-function for the weight vector 11111 for the variables
res-vlist.  cf. wbf
Example: [ [((x Dx -1 ) x Dx (x Dx + 2)) (y Dy)] (x,y) (x)] bf-111
```

10. bfunction

```
 a bfunction b
array a; poly b;
a :  [f] ;  string f ;
a :  [f] ;  polynomial f ;
a :  [f v] ; string f,v;
a :  [f v] ; polynomial f, string v;
b is the b-function (=Bernstein-Sato polynomial) of a polynomial f
in variables v.
If v is not specified, the variables are assumed to be (x,y,z).
b will be a polynomial in s.  This variable can be changed by typing in
  (variable) /bfunction.s set
For the algorithm, see Duke Math. J. 87 (1997),115-132,
   J. Pure and Applied Algebra 117&118(1997), 495--518.
Example  [(x^3-y^2) (x,y)] bfunction ::
```

11. ch

```
ch is the abbreviation of characteristic.
 a ch b
 array a;  number b
b is the generator of the characteristic variety of a.
For the algorithm, see, Japan J. of Industrial and Applied Math., 1994, 485--497.
Example 1 :
 [ [( (x Dx)^2 + ( y Dy)^2) ( x Dx y Dy -1)] (x,y)] ch ::
Example 2 :
[ [( (x^3-y^2) Dx + 3 x^2) ( (x^3-y^2) Dy - 2 y)] (x,y)] ch ::
```

12. characteristic

```
 a characteristic b
 array a;  number b
b is the generator of the characteristic variety of a.
```

```
For the algorithm, see Japan J. of Industrial and Applied Math., 1994, 485--497.
Example 1 :
 [ [( (x Dx)^2 + ( y Dy)^2) ( x Dx y Dy -1)] (x,y)] characteristic ::
Example 2 :
[ [( (x^3-y^2) Dx + 3 x^2) ( (x^3-y^2) Dy - 2 y)] (x,y)] characteristic ::
```

13. `db.clear`

```
db.clear cleans db.VariableStack
cf.  db.restore, db.where, db.clear.ds, db.clear.es, debugMode
```

14. `db.clear.ds`

```
db.clear.ds cleans db.DebugStack
cf.  db.where.ds, db.clear, debugMode
```

15. `db.clear.es`

```
db.clear.es cleans db.ErrorStack
cf.  db.clear, db.where.es
```

16. `db.clear.es`

```
db.clear.es cleans db.ErrorStack
cf.  db.clear, db.where.es
```

17. `db.restore`

```
db.restore recovers bindings of variables by reading db.VariableStack
cf. localVariables, restoreVariables,
    db.clear, db.where , debugMode
```

18. `db.where`

```
db.where shows the db.VariableStack
cf. localVariables, restoreVariables,
    db.clear, db.restore, db.where.ds, db.where.es, debugMode
```

19. `db.where.es`

```
db.where.es shows the db.ErrorStack
Error and warning messages are put in db.ErrorStack when the global
variables ErrorMessageMode or WarningMessageMode are set to 1 or 2.
cf. db.where, system_variable
```

20. deRham

```
[f v] deRham c
string f; string v;  f is a polynomial given by a string.
This function can be used by loading the experimental package cohom.sm1.
The dimensions of the deRham cohomology groups H^i(C^n - V(f),C) i=0, i=1, ..
.., n are returned in c.
For example, if c=[1 4 6 4], then it means that dim H^0(C^3-V(f),C) = 1,
dim H^1(C^3-V(f),C) = 4, and so on.
For the algorithm, see "An algorithm for de Rham cohomology groups of the
complement of an affine variety via D-module computation",
Journal of pure and applied algebra, 139 (1999), 201--233. math.AG/9801114

Example 0:  [(x (x-1) (x-2)) (x)] deRham
Example 1: [(x y (x+y-1)(x-2)) (x,y)] deRham
Example 2: [(x^3-y^2) (x,y)] deRham
Example 3: [(x^3-y^2 z^2) (x,y,z)] deRham
Example 4: [(x y z (x+y+z-1)) (x,y,z)] deRham
```

21. debugMode

```
debugMode overrides on the functions pushVariables and popVariables
and enables to use db.where.  Never execute debugMode inside a block of
pushVariables and popVariables
```

22. fctr

```
The keyword <<fctr>> cannot be found.
Type in ? in order to see system dictionary.
```

23. gb

```
a gb b
array a; array b;
b : [g ii];  array g; array in; g is a Grobner basis of f
             in the ring of differential operators.
            ii is the initial ideal in case of w is given or <<a>> belongs
            to a ring. In the other cases, it returns the initial monominal.
a : [f ];    array f;  f is a set of generators of an ideal in a ring.
a : [f v];   array f; string v;  v is the variables.
a : [f v w]; array f; string v; array of array w; w is the weight matirx.
```

```
Example 1: [ [( (x Dx)^2 + (y Dy)^2 -1) ( x y Dx Dy -1)] (x,y)
             [ [ (Dx) 1 ] ] ] gb pmat ;
Example 2:
To put h=1, type in, e.g.,
 [ [(2 x Dx + 3 y Dy+6) (2 y Dx + 3 x^2 Dy)] (x,y)
   [[(x) -1 (Dx) 1 (y) -1 (Dy) 1]]] gb /gg set gg dehomogenize pmat ;

Example 3: [ [( (x Dx)^2 + (y Dy)^2 -1) (  x y Dx Dy -1)] (x,y)
             [ [ (Dx) 1 (Dy) 1] ] ] gb pmat ;

Example 4: [[ [(x^2) (y+x)] [(x+y) (y^3)] [(2 x^2+x y) (y+x+x y^3)]] (x,y)
             [ [ (x) -1 (y) -1] ] ] gb pmat ;

cf. gb, groebner, groebner_sugar, syz.
```

24. genericAnn

```
f [s v1 v2 ... vn] genericAnn [L1 ... Lm]
L1, ..., Lm are annihilating ideal for f^s.
f is a polynomial of v1, ..., vn
<string> | <poly>  f, s, v1, ..., vn ; <poly>  L1, ..., Lm
Example: (x^3+y^3+z^3) [(s) (x) (y) (z)] genericAnn
```

25. genericAnnWithL

```
[f v] genericAnnWithL [b [L I]]
String f,v; poly b,L; array of poly I;
f is a polynomial given by a string. v is the variables.
 v must not contain names  s, e.
b is the b-function (Bernstein-Sato polynomial) for f and
 L is the operator satisfying L f^{s+1} = b(s) f^s
 I is the annihilating ideal of f^s.
cf. bfunction, annfs, genericAnn.
Example 1:  [(x^2+y^2) (x,y)] genericAnnWithL ::
Example 2:  [(x^2+y^2+z^2) (x,y,z)] genericAnnWithL ::
Example 3:  [(x^3-y^2 z^2) (x,y,z)] genericAnnWithL ::
```

26. gkz

```
[ A b] gkz [eq v]
[ A  ] gkz [eq v]
[    ] gkz [eq v]
array of array of integer A; array of integer b;
eq is the GKZ system defined by the matrix A and the parameter b.
v is the list of variables.
Default values of A and b are in gkz.A and gkz.b
For details, see Functional analysis and its applications, 23, 1989, 94--106.
    Grobner deformations of hypergeometric differential equations, Springer, 1999
Example 1: [ [[1 1 1 1] [0 1 3 4]] [1 2]] gkz rrank ::
Example 2: [ [[1 1 1 1] [0 1 3 4]] [0 0]] gkz rrank ::
```

27. integral-k1

```
[[f1 ... fm] [v1 ... vn] [v1 w1 ... vp wp] k1] integral0
                                              [[g1 ... gq],[e1,...,er]]
poly|string f1 ...fm; string v1 ... vn;
string v1 ... vp; integer w1 ... wp;
integer k1;
poly g1 ... gq; poly e1, ..., er;
f1 ... fm are annihilors, v1 ... vn are variables,
w1 is the weight of the variable v1, ...
k1 is the maximal degree of the filtration: maximal integral root
of b-function. cf. intwbf
g1, ..., gq are integral. e1, ..., er are basis of the free module to which
the g1, ..., gq belong.
THE ORDERS OF INTEGRAL VARIABLES MUST BE SAME BOTH IN THE SECOND AND
THE THIRD ARGUMENTS. INTEGRAL VARIABLES MUST APPEAR FIRST.
Example 1: [[(x-y) (Dx+Dy)] [(y) (x)] [(y) -1 (Dy) 1] 1] integral-k1
Example 2: [[(x (x-1)) (x)] annfs 0 get [(x)] [(x) -1 (Dx) 1] 1] integral-k1
```

28. integration

```
 [[f1 f2 ...] [t1 t2 ...] [vars params] [k0 k1 limitdeg ]] integration
  [ 0-th cohomology group,  (-1)-th cohomology group, .... ]

 [[f1 f2 ...] [t1 t2 ...] [vars params] limitdeg] integration

This function can be used by loading the experimental package cohom.sm1.
Integration of the D-ideal [f1 f2 ...] to t1=0, t2=0, ... is computed.
vars is a list of the variables and params is a list of parameters.
k0 is the minimum integral root of the b-function and k1 is the maximum
integral root of the b-function. If these values are not given and
they are small, then they are automatically computed. The program returns
 0-th, ..., -limitdeg-th cohomology groups.
[vars params] and [k0 k1 deg] are optional arguments.
If vars and params are not given, the values of the global variables
BFvarlist and BFparlist will be used.
The operator restriciton will be used after the laplace transformation.

For the algorithm, see math.AG/9805006, http://xxx.langl.gov

Example 1:
[[(x (x-1)) (x) ] annfs 0 get       [(x)] [[(x)] [ ]]] integration ::
Example 2:
[ [(Dt - (3 t^2-x)) (Dx + t)] [(t)]      [[(t) (x)] [ ]] 0] integration ::
Example 3:
[ [[(Dt - (3 t^2-x)) (0)] [ (Dx + t) (0)]] [(t)]      [[(t) (x)] [ ]] 0] integration ::
In case of vector input, INTEGRAL VARIABLES MUST APPEAR FIRST
in the list of variable. We are using wbfRoots to get the roots of
b-functions, so we can use only generic weight vector for now.
```

29. intersection

```
Ideal intersections in the ring of differential operators.
Example 1: [[[(x1) (x2)] [(x2) (x4)] (x1,x2,x3,x4)] intersection
            [(x2) (x4^2)] (x1,x2,x3,x4)] intersection ::
Example 2: [[[(x1) (x2)] [(x2) (x4)] (x1,x2,x3,x4)] intersection
            [(x2) (x4^2)] (x1,x2,x3,x4)] intersection /ff set ff message
           [ ff [(x2^2) (x3) (x4)] (x1,x2,x3,x4)] intersection ::
Example 3: [[[(x1) (x2)] [(x2) (x4^2)] (x1,x2,x3,x4)] intersection
            [(x2^2) (x3) (x4)] (x1,x2,x3,x4)] intersection ::
```

30. intwbf

```
[[f1 ... fm] [v1 ... vn] [v1 w1 ... vp wp]] intwbf [g1 ... gq]
<poly>|<string> f1 ...fm; <string> v1 ... vn;
<string> v1 ... vp; <integer> w1 ... wp;
<poly> g1 ... gq;
f1 ... fm are generators, v1 ... vn are variables,
w1 is the weight of the variable v1, ...
THE ORDERS OF INTEGRAL VARIABLES MUST BE SAME BOTH IN THE SECOND AND
THE THIRD ARGUMENTS. INTEGRAL VARIABLES MUST APPEAR FIRST.
If the weight is not generic, then the function exits with error.
Example 1: [[(x-y) (Dx+Dy)] [(y) (x)] [(y) -1 (Dy) 1]] intwbf
           integrate only for y.
Example 2: [[(-3 x^2 Dy-2 y Dx) (2 x Dx+3 y Dy+6)] [(x) (y)]
           [(x) -1 (Dx) 1 (y) -2 (Dy) 2]] intwbf
Example 3: [[[(0) (x^2 Dx+x)] [(Dx^2+x Dx^3) (0)]] [(x)]
           [(x) -1 (Dx) 1]] intwbf
```

31. intwbfRoots

```
This function needs oxasir  --- rationalRoots2
This function is defined in intw.sm1 and requires oxasir.sm1 and ox_asir server.
Example 1:  [[(-3 x^2 Dy-2 y Dx) (2 x Dx+3 y Dy+6)] [(x) (y)]
            [(x) -1 (Dx) 1 (y) -2 (Dy) 2]] intwbfRoots
Example 2:   [[[(0) (x^2 Dx+x)] [(Dx^2+x Dx^3) (0)]] [(x)]
              [(x) -1 (Dx) 1]] intwbfRoots
```

32. loadAsirFile

```
The keyword <<loadAsirFile>> cannot be found.
Type in ? in order to see system dictionary.
```

33. localVariables

```
This function is as same as pushVariables, but it pushes the variable to
db.VariableStack
cf. db.where, pushVariables, restoreVariables, debugMode
```

34. ox

```
client aa oxpushcmo (oxsendcmo), client oxpopcmo dd, client popstring dd, client ss oxrpc
Example: ox.ccc [1 2 3] oxpushcmo ox.ccc oxpopcmo /ff set  ff ::

client ss oxexecutestring(oxsubmit) <<executeString>>, client ss oxexec <<executeFunction>>
Example: ox.ccc (0 1 10 { pop $Hello$ message } for) oxexecutestring
Example: ox.ccc (2).. oxpushcmo ox.ccc (10).. oxpushcmo
         ox.ccc (power) oxexec  ox.ccc oxpopcmo ::

client str oxevalname, client str oxsetname, client num oxpops, client oxget

client oxmathcap, client oxsetmathcap, oxgenportfile, oxremoveportfile

client oxgetsp --- get stack pointer,  client oxgeterrors --- get all errors
cleint oxisData, client oxreset, client oxshutdown
oxhelp for this message, [(oxWatch) client] extension to start a log.
[(oxSerial)] extension serial-number-of-out-going-ox-packet.
oxpopcmo1, oxpopcmo2,  sm1connect2, sm1connect3
sm1connectr ox_launch, ox_launch_nox
oxNoX : set an environment to invoke oxlog without xterm and to call oxlog with >& /dev/null
        ex 1.  (ox.sm1) run oxNoX sm1connectr
        ex 2.  (oxasir.sm1) run oxNoX (x^2-1). fctr pmat ;
```


```
client aa oxpushcmo (oxsendcmo), client oxpopcmo dd, client popstring dd, client ss oxrpc
Example: ox.ccc [1 2 3] oxpushcmo ox.ccc oxpopcmo /ff set  ff ::

client ss oxexecutestring(oxsubmit) <<executeString>>, client ss oxexec <<executeFunction>>
Example: ox.ccc (0 1 10 { pop $Hello$ message } for) oxexecutestring
Example: ox.ccc (2).. oxpushcmo ox.ccc (10).. oxpushcmo
         ox.ccc (power) oxexec  ox.ccc oxpopcmo ::

client str oxevalname, client str oxsetname, client num oxpops, client oxget

client oxmathcap, client oxsetmathcap, oxgenportfile, oxremoveportfile

client oxgetsp --- get stack pointer,  client oxgeterrors --- get all errors
cleint oxisData, client oxreset, client oxshutdown
oxhelp for this message, [(oxWatch) client] extension to start a log.
[(oxSerial)] extension serial-number-of-out-going-ox-packet.
oxpopcmo1, oxpopcmo2,  sm1connect2, sm1connect3
sm1connectr ox_launch, ox_launch_nox
oxNoX : set an environment to invoke oxlog without xterm and to call oxlog with >& /dev/null
        ex 1.  (ox.sm1) run oxNoX sm1connectr
        ex 2.  (oxasir.sm1) run oxNoX (x^2-1). fctr pmat ;
```


35. ox_launch

```
 [remote remote-OpenXM-bin remote-ox-server remote-login-name myhostname]
  ox_launch  client
or
 [(localhost) ox-server-name] ox_launch client
Example 1:
 [(tau.math.kobe-u.ac.jp) (/home/taka/OpenXM/bin)
  (/home/taka/OpenXM/src/ox_math/ox_math) (taka) (dc4.math.kobe-u.ac.jp)
```

```
 ] ox_launch /@@@.oxmath set
 [remote remote-OpenXM-bin remote-ox-server remote-login-name myhostname]
  ox_launch  client
or
 [(localhost) ox-server-name] ox_launch client
Example 1:
 [(tau.math.kobe-u.ac.jp) (/home/taka/OpenXM/bin)
  (/home/taka/OpenXM/src/ox_math/ox_math) (taka) (dc4.math.kobe-u.ac.jp)
 ] ox_launch /@@@.oxmath set
```

36. ox_launch_nox

```
 [remote remote-OpenXM-bin remote-ox-server remote-login-name myhostname]
  ox_launch_nox  client
Example 1:
 [(tau.math.kobe-u.ac.jp) (/home/taka/OpenXM/bin)
  (/home/taka/OpenXM/src/ox_math/ox_math) (taka) (dc4.math.kobe-u.ac.jp)
 ] ox_launch_nox /@@@.oxmath set
 [remote remote-OpenXM-bin remote-ox-server remote-login-name myhostname]
  ox_launch_nox  client
Example 1:
 [(tau.math.kobe-u.ac.jp) (/home/taka/OpenXM/bin)
  (/home/taka/OpenXM/src/ox_math/ox_math) (taka) (dc4.math.kobe-u.ac.jp)
 ] ox_launch_nox /@@@.oxmath set
```

37. oxconnect

```
[hostname data-port control-port] oxconnect pid
oxconnect is used to connect this to a server on the hostname with data-port
and control-port
Example 1:  ox -ox ox_sm1 -data 1300 -control 1200    <== on server.
            [$localhost$ 1300 1200] oxconnect /ox.ccc set  <== on client.
Example 2:  ox -ox ox_sm1 -host s.OpenXM.org -insecure -data 1300 -control 1200    <== on server.
            [$s.OpenXM.org$ 1300 1200] oxconnect /ox.ccc set  <== on client.
Example 3:  ox -ox ox_sm1 -data 1300 -control 1200  >& /dev/null &<== on server.
            [$localhost$ 1300 1200] oxconnect /ox.ccc set  <== on client.
           cf. oxNoX
See also ox
[hostname data-port control-port] oxconnect pid
oxconnect is used to connect this to a server on the hostname with data-port
and control-port
Example 1:  ox -ox ox_sm1 -data 1300 -control 1200    <== on server.
            [$localhost$ 1300 1200] oxconnect /ox.ccc set  <== on client.
Example 2:  ox -ox ox_sm1 -host s.OpenXM.org -insecure -data 1300 -control 1200    <== on server.
            [$s.OpenXM.org$ 1300 1200] oxconnect /ox.ccc set  <== on client.
Example 3:  ox -ox ox_sm1 -data 1300 -control 1200  >& /dev/null &<== on server.
            [$localhost$ 1300 1200] oxconnect /ox.ccc set  <== on client.
           cf. oxNoX
See also ox
```

38. pgb

```
a pgb b
array a; array b;
b : [g ii];  array g; array in; g is a Grobner basis of f
             in the ring of polynomials.
             ii is the initial ideal in case of w is given or <<a>>belongs
             to a ring. In the other cases, it returns the initial monominal.
a : [f ];    array f;  f is a set of generators of an ideal in a ring.
a : [f v];   array f; string v;  v is the variables.
a : [f v w]; array f; string v; array of array w; w is the weight matirx.
Example 1: [(x,y) ring_of_polynomials 0] define_ring
           [ [(x^2+y^2-4). (x y -1).] ] pgb ::
Example 2: [ [(x^2+y^2) (x y)]   (x,y)  [ [(x) -1 (y) -1] ] ] pgb ::
cf. gb, groebner, groebner_sugar, syz.
```

39. primadec

```
The keyword <<primadec>> cannot be found.
Type in ? in order to see system dictionary.
```

40. ptozp_subst

```
The keyword <<ptozp_subst>> cannot be found.
Type in ? in order to see system dictionary.
```

41. rank

```
 a rank b
 array a;  number b
Example 1 :
 [ [( (x Dx)^2 + ( y Dy)^2) ( x Dx y Dy -1)] (x,y)] rank ::
Example 2 :
[ [( (x^3-y^2) Dx + 3 x^2) ( (x^3-y^2) Dy - 2 y)] (x,y)] rank ::
```

42. reduceByAsir

```
The keyword <<reduceByAsir>> cannot be found.
Type in ? in order to see system dictionary.
```

43. res-div

```
 [M N] res-div K
 matrix M, N, K ; Each element of M and N must be an element of a ring.
 coker(K) is isomorphic to M/N.
```

```
Example: [(x,y) ring_of_differential_operators 0] define_ring
   [[[(x+x^2+y^2).] [(x y).]] [[(x+x^2+y^2).] [(x y).]]] res-div

res*div accepts string inputs, too. For example,
 [[[[(x+x^2+y^2)] [(x y)]] [[(x+x^2+y^2)] [(x y)]]]
   [(x) (y)]]  res*div ::
See also res-toString, res-setRing.
```

44. res-dual

```
[F V] res-dual G
G is the dual D-module of F. V is a list of variables.
Example 1:  [ [( x^3-y^2 )  ( 2 x Dx + 3 y Dy + 6 )  ( 2 y Dx + 3 x^2 Dy) ]
              [(x) (y)]] res-dual
Example 2:  [[1 3 4 5]] appell1 res-dual
Example 3:  [ [(-x1 Dx1 + x1 + 2) (x2 Dx2 - Dx2 -3)] [(x1) (x2)]] res-dual
Example 4:  [ [(x2 Dx2 - Dx2 + 4) (x1 Dx1 + x1 +3)] [(x1) (x2)]] res-dual
            3 and 4 are res-dual each other.
Example 5:  [ [[1 1 1][0 1 2]] [0 0]] gkz res-dual
Example 6:  [ [[1 1 1][0 1 2]] [-2 -1]] gkz res-dual

Example 7:  [ [(x Dx -1) (Dx^2)]     [(x)]] res-dual
Example 8:  [ [[(1) (0)] [(0) (Dx)]] [(x)]] res-dual
Example 9:  [ [((x Dx + x +1) (Dx-1))] [(x)]] res-dual
```

45. res-kernel-image

```
 [m n vlist] res-kernel-image c
When, D^p <-- m --- D^q <-- n -- D^r
D^q/c is isomorhic to ker(m)/im(n).
vlist is a list of variables.
```

46. res-solv

```
[M d] res-solv [c' r]
 M : matrix,  d, c' : vectors, r : scalar(integer)
 c:=c'/r is a solutions of Sum[c_i M_i] = d where c_i is the i-th element
 of the vector c and M_i is the i-th row vector of M.
If there is no solution, then res-solv returns null.
Note that M and d are not treated as an element of the homogenized Weyl
algebra. If M or d contains the homogenization variable h, it automatically
set to 1. If you need to use h, use the command res-solv-h
Example 1:  [(x,y) ring_of_differential_operators [[(x) -1 (Dx) 1]] weight_vector 0]
             define_ring
 [ [ [(x Dx + 2).] [ (Dx (x Dx + 3) - (x Dx + 2) (x Dx -4)).]]   [(1).]]
  res-solv ::
Example 2:
 [ [ (x Dx + 2).  (Dx (x Dx + 3) - (x Dx + 2) (x Dx -4)).]   (1).]
  res-solv ::
Example 3:
```

```
 [ [[(x Dx + 2). (0).]
    [(Dx+3).     (x^3).]
    [(3).        (x).]
    [(Dx (x Dx + 3) - (x Dx + 2) (x Dx -4)). (0).]]   [(1). (0).]]
  res-solv ::
Example 4:
 [[ (x*Dx+h^2). (Dx^2+x*h).] [(x^2+h^2). (h Dx + x^2).]] /ff set
 [[ (x^2 Dx + x h^2). (Dx^3).]] /gg set
 [ff gg ff mul 0 get ] res-solv-h ::

res*solv and res*solv*h accept string inputs, too. For example,
 [[ [ [(x Dx + 2)] [ (Dx (x Dx + 3) - (x Dx + 2) (x Dx -4))]]   [(1)]]
  (x)]  res*solv ::
```

47. resol0

```
 [ ii v] resol0 r
array of poly ii; string v;
<< vv >> is a string of variables separated by ,

 [ ii v] resol0 r
array of poly ii; array of strings v;
<< vv >> is an array of variable names.

 [ ii v w] resol0 r
array of poly ii; string v; array w;
<< w >> is a weight vector.

You can also give a parameter << d >> to specify the truncation depth
of the resolution: [ d ii v] resol0, [d ii v w] resol0

resol0 constructs a resolution which is adapted (strict)
to a filtration. So, it is not minimal.
 r = [starting Groebner basis g, [ s1, s2 , s3, ...], order-def].
g is the reduced Groebner basis for f,
 s1 is the syzygy of g,
 s2 is the syzygy of s1,
 s3 is the syzygy of s2 and so on.
For details, see math.AG/9805006
cf. sResolution, tparse, s_ring_..., resol0.cp
Example:  [ [( x^3-y^2 )  ( 2 x Dx + 3 y Dy + 6 )  ( 2 y Dx + 3 x^2 Dy) ]
             (x,y) ] resol0 ::
```

48. resol1

```
 [ ii v] resol1 r
array of poly ii; string v;
<< vv >> is a string of variables separated by ,

 [ ii v] resol1 r
array of poly ii; array of strings v;
<< vv >> is an array of variable names.
```

```
 [ ii v w] resol1 r
array of poly ii; string v; array w;
<< w >> is a weight vector.

 ii may be array of array of poly.
You can also give a parameter << d >> to specify the truncation depth
of the resolution: [ d ii v] resol1, [d ii v w] resol1

resol1 constructs a resolution which is adapted (strict)
to a filtration. So, it is not minimal in general.
 r = [s0, s1, s2 , s3, ...].
 s0 is the groebner basis of ii,
 s1 is the syzygy of s0,
 s2 is the syzygy of s1,
 s3 is the syzygy of s2 and so on.
 s1 s0 mul ==> 0, s2 s1 mul ==>0, ...
For details, see math.AG/9805006
cf. sResolution, tparse, s_ring_..., resol0.cp
resol1.withZeroMap returns a resolution with zero maps of the both sides
   of the resolution.
cf. resol1.zeroMapL, resol1.zeroMapR, resol1.withZeroMap.aux
resol1.syzlist : global variable to keep the raw output of sResolution.

Example 1: [ [( x^3-y^2 )  ( 2 x Dx + 3 y Dy + 6 )  ( 2 y Dx + 3 x^2 Dy) ]
             (x,y) ] resol1 pmat ;
Example 2: [ [( x^3-y^2 )  ( 2 x Dx + 3 y Dy + 6 )  ( 2 y Dx + 3 x^2 Dy) ]
             (x,y) [[(x) -1 (Dx) 1 (y) -1 (Dy) 1]]] resol1 pmat ;
Example 3: [ [[(2 x Dx + 3 y Dy +6) (0)]
              [(3 x^2 Dy + 2 y Dx) (0)]
              [(0) (x^2+y^2)]
              [(0) (x y )] ]
             (x,y) [[(x) -1 (Dx) 1 (y) -1 (Dy) 1]]] resol1 pmat ;
Example 4:  /resol0.verbose 1 def
          [ [[(x^2+y^2+ x y) (x+y)] [(x y ) ( x^2 + x y^3)] ] (x,y)
            [[(x) -1 (Dx) 1 (y) -1 (Dy) 1]]] resol1 pmat ;
```

49. restall_s

```
f v s_0 s_1 level restall_s c
array f; array v; integer s_0, s_1, level; array c
f is a set of generators. v is a set of variables.
s_0 is the minimal integral root of the b-function.
s_1 is the maximal integral root of the b-function.
level is the truncation degree of the resolution.
c is the cohomolgy.
This command is vector clean for Schreyer 1 and 2.

Global variables: array BFvarlist : list of variables.
                  array BFparlist : list of parameters.
                  int BFmessage : verbose or not.
                  int Schreyer : Schreyer 1 with tower-sugar.sm1 -- V-homog.
                                 Schreyer 2 with tower.sm1 --- H-homog.
                                 Schreyer 0 is a step by step resolution.
result global var:array BFresolution : resolution matrices.
```

```
cf. /tower.verbose 1 def /tower-sugar.verbose 1 def
    /debug.sResolution 1 def
       to see steps of resoluitons.
  /Schreyer 2 def  (tower.sm1) run   to try Schreyer 2. Schreyer 2 is default.
See bfm --- b-function, restriction.
restall1_s is as same as restall_s, except it does not truncate from below.
Example 1:  /BFvarlist [(x1) (x2)] def /BFparlist [ ] def
            [(x1) (Dx2-1)] [(x1)] -1 -1 1 restall_s
Example 1a: /BFvarlist [(x1) (x2)] def /BFparlist [ ] def
            [(x1) (Dx2-1)] [(x1)] bfm
Example 2:  /BFvarlist [(x) (y)] def /BFparlist [ ] def
 [[(x Dx -1) (0)] [(y Dy -2) (0)]  [(1) (1)] ] /ff set
 ff [(x) (y)] 0 4 2 restall_s
```

50. restoreVariables

```
This function is as same as popVariables, but it pops the variable from
db.VariableStack
cf. db.where, popVariables, localVariables, debugMode
```

51. restriction

```
 [[f1 f2 ...] [t1 t2 ...] [vars params] [k0 k1 limitdeg ]] restriction
  [ 0-th cohomology group,  (-1)-th cohomology group, .... ]

 [[f1 f2 ...] [t1 t2 ...] [vars params] limitdeg] restriction

This function can be used by loading the experimental package cohom.sm1.
Restriction of the D-ideal [f1 f2 ...] to t1=0, t2=0, ... is computed.
vars is a list of the variables and params is a list of parameters.
k0 is the minimum integral root of the b-function and k1 is the maximum
integral root of the b-function. If these values are not given and
they are small, then they are automatically computed. The program returns
 0-th, ..., -limitdeg-th cohomology groups.
[vars params] and [k0 k1 deg] are optional arguments.
If vars and params are not given, the values of the global variables
BFvarlist and BFparlist will be used.

For the algorithm, see math.AG/9805006, http://xxx.langl.gov

Example 1: cf. math.AG/9801114, Example 1.4
  [[(- 2 x Dx - 3 y Dy +1) (3 y Dx^2 - 2 x Dy)]
     [(x) (y)] [[(x) (y)] [ ]]] restriction  ::
[    [    0 , [   ] ]  , [    1 , [   ] ]  , [    1 , [   ] ] ]
       H^0 = 0,     H^(-1)= C^1/(no relation), H^(-2)=C^1/(no relation).
Example 2:
[[(x Dx-1) (Dy^2)] [(y)] [[(x) (y)] [ ]]] restriction ::
[    [    2 , [    -x*Dx+1 , -x*e*Dx+e ]  ]  , [    0 , [   ] ] ]
     H^0=D_1^2/([-x Dx+1,0],[0, -x Dx + 1]),  H^(-1) = 0
     where  e^0, e^1, e^2, ..., e^(m-1) are standard basis vectors of
            rank m free module (D_1)^m. D_1 is the ring of differential
            opertors of one variable x.
Example 3:
```

```
[[(x Dx-1) (Dy^2)] [(y)] [[(x) (y)] [ ]] 0] restriction ::
Example 4:
[[[(0) (x^2 Dx+x)] [(Dx^2+x Dx^3) (0)]] [(x)] [[(x)] [ ]]] restriction ::
In case of vector input,  RESTRICTION VARIABLES MUST APPEAR FIRST
in the list of variable. We are using wbfRoots to get the roots of
b-functions, so we can use only generic weight vector for now.
```

52. rrank

```
 a rrank b
 array a;  number b
It computes the holonomic rank for regular holonomic system.
For the algorithm, see Grobner deformations of hypergeometric differential equations, 1999, Springer.
Chapter 2.
Example 1 :
 [ [( (x Dx)^2 + ( y Dy)^2) ( x Dx y Dy -1)] (x,y)] rrank ::
```

53. saturation

```
[ideal J vlist] saturations jjj
It returns (ideal : J^\infty)
Saturation is computed in the ring of polynomials.
When J=[f_1, f_2, ...], it is equal to
((ideal, z-(f_1 + y f_2 + y^2 f_3 +...)) : z^\infty) \cap k[x].
Example 1:
           [[(x1 y1 + x2 y2 + x3 y3 + x4 y4)
             (x2 y2 + x4 y4) (x3 y3 + x4 y4) (y1 y4 - y2 y3)]
            [(y1) (y2) (y3) (y4)] (x1,x2,x3,x4,y1,y2,y3,y4)] saturation
            /ff set [ff (x1,x2,x3,x4,y1,y2,y3,y4)
                     [[(y1) 1 (y2) 1 (y3) 1 (y4) 1]]] pgb
            0 get [(y1) (y2) (y3) (y4)] eliminatev ::
Example 2: [[(x2^2) (x2 x4) (x2) (x4^2)] [(x2) (x4)] (x2,x4)] saturation
```

54. syz

```
a syz [b c]
array a; array b; array c
b is a set of generators of the syzygies of f.
c = [gb, backward transformation, syzygy without dehomogenization].
See groebner.
a : [f ];    array f;  f is a set of generators of an ideal in a ring.
a : [f v];   array f; string v;  v is the variables.
a : [f v w]; array f; string v; array of array w; w is the weight matirx.
Example 1: [(x,y) ring_of_polynomials 0] define_ring
           [ [(x^2+y^2-4). (x y -1).] ] syz ::
Example 2: [ [(x^2+y^2) (x y)]   (x,y)  [ [(x) -1 (y) -1] ] ] syz ::
Example 3: [ [( (x Dx)^2 + (y Dy)^2 -1) ( x y Dx Dy -1)] (x,y)
             [ [ (Dx) 1 ] ] ] syz pmat ;
Example 4:  [ [(2 x Dx + 3 y Dy+6) (2 y Dx + 3 x^2 Dy)] (x,y)
             [[(x) -1 (Dx) 1 (y) -1 (Dy) 1]]] syz pmat ;
```

```
Example 5:  [ [ [(x^2) (y+x)] [(x+y) (y^3)] [(2 x^2+x y) (y+x+x y^3)]]
              (x,y) ] syz pmat ;
Example 6:  [ [ [(x^2) (y+x)] [(x+y) (y^3)] [(2 x^2+x y) (y+x+x y^3)]]
              (x,y) [[(x) -1 (y) -2]] ] syz pmat ;
Example 7:  [ [ [(0) (0)] [(0) (0)] [(x) (y)]]
              [(x) (y)]] syz pmat ;
```

55. tensor0

```
 [F G vlist] tensor0
This function requires the package cohom.sm1.
Example 1:
 [[(2 x Dx - 1)] [(2 x Dx - 3)] (x)] tensor0
Example 2:
 [[(-x*Dx^2+x-Dx+1)] [((x Dx + x +1)(Dx-1))] (x)] tensor0
Example 3:
 [[(x Dx -1) (y Dy -4)] [(Dx + Dy) (Dx-Dy^2)] (x,y)] tensor0
```

56. wIntegration0

```
 [gg vlist weight] wIntegration0  [ igg bb]
list of strings gg; list of strings vlist;
list weight;
integer k1;
list of polys igg; list of polys base;
gg are input ideal or submodule.
igg are relations and bb are bases. They give the integral.
This function fails when weight is not generic.
cf. intwbf, intwbfRoots, integral-k1.
This function is defined in intw.sm1 and requires oxasir.sm1 and ox_asir server.
See Grobner Deformations of Hypergeometric Differential Equations, Springer
    Section 5.5 for the algorithm.
Example 1: [ [(Dt - (3 t^2-x)) (Dx + t)] [(t) (x)] [(t) -1 (Dt) 1]]
              wIntegration0
Example 2: [[(-3 x^2 Dy-2 y Dx) (2 x Dx+3 y Dy+6)] [(x) (y)]
             [(x) -1 (Dx) 1 (y) -2 (Dy) 2]]  wIntegration0
           The output [[-x, 1] [x,1]] implies the integral is
           (K x + K 1)/(K (-x) + K 1) = 0 where K is the base field and
           x and 1 is the vector space basis.
           Note that the order of weight and the order of the variables
           must be the same.  Note also that the next of (x) must be (Dx)
           and so on.
```

57. wRestriction0

```
 [gg vlist weight] wRestriction0  [ igg bb]
list of strings gg; list of strings vlist;
list weight;
integer k1;
list of polys igg; list of polys base;
```

```
gg are input ideal or submodule.
igg are relations and bb are bases. They give the 0-th restriction.
This function fails when weight is not generic.
cf. intwbf, intwbfRoots, integral-k1.
This function is defined in intw.sm1 and requires oxasir.sm1 and ox_asir server.
See Grobner Deformations of Hypergeometric Differential Equations, Springer
    Section 5.5 for the algorithm.
Example 1: [ [(Dt^2) (Dx^2)] [(t) (x)] [(t) -1 (Dt) 1]]
             wRestriction0
Example 2: [[(Dx^2) (Dy^2)] [(x) (y)]
            [(x) -1 (Dx) 1 (y) -2 (Dy) 2]]  wRestriction0
           The output [[-Dx, 1] [Dx,1]] implies the restriction is
           (K Dx + K 1)/(K (-Dx) + K 1) = 0 where K is the base field and
           Dx and 1 is the vector space basis.
           Note that the order of weight and the order of the variables
           must be the same.  Note also that the next of (x) must be (Dx)
           and so on.
```

58. wTensor0

```
[F G v weight] wTensor0
See tensor0
It calls wRestriction0 instead of restriction.
Example 1:
 [[(x Dx -1) (y Dy -4)] [(Dx + Dy) (Dx-Dy^2)] (x,y) [1 2]] wTensor0
```

59. wbf

```
[[f1 ... fm] [v1 ... vn] [v1 w1 ... vp wp]] wbf [g1 ... gq]
<poly>|<string> f1 ...fm; <string> v1 ... vn;
<string> v1 ... vp; <integer> w1 ... wp;
<poly> g1 ... gq;
f1 ... fm are generators, v1 ... vn are variables,
w1 is the weight of the variable v1, ...
THE ORDERS OF INTEGRAL VARIABLES MUST BE SAME BOTH IN THE SECOND AND
THE THIRD ARGUMENTS. INTEGRAL VARIABLES MUST APPEAR FIRST.
If the weight is not generic, then the function exits with error.
cf. bf-111 for w=(1 1 1 1 ...)
Example 1: [[(x-y) (Dx+Dy)] [(y) (x)] [(y) -1 (Dy) 1]] wbf
            restrict only for y.
Example 2: [[(-3 x^2 Dy-2 y Dx) (2 x Dx+3 y Dy+6)] [(x) (y)]
            [(x) -1 (Dx) 1 (y) -2 (Dy) 2]] wbf
Example 3: [[[(0) (x^2 Dx+x)] [(Dx^2+x Dx^3) (0)]] [(x)]
            [(x) -1 (Dx) 1]] wbf
```

60. wbfRoots

```
This function needs oxasir  --- rationalRoots2
This function is defined in intw.sm1 and requires oxasir.sm1 and ox_asir server.
Example 1:  [[(-3 x^2 Dy-2 y Dx) (2 x Dx+3 y Dy+6)] [(x) (y)]
```

```
             [(x) -1 (Dx) 1 (y) -2 (Dy) 2]] wbfRoots
Example 2:   [[[(0) (x^2 Dx+x)] [(Dx^2+x Dx^3) (0)]] [(x)]
               [(x) -1 (Dx) 1]] wbfRoots
```

61. `wdeRham0`

```
It computes the midle dimensional cohomology groups and bases.
A generic weight vector is used for the computation.
This function is defined in intw.sm1 and requires oxasir.sm1 and ox_asir server.
 Example 1 : [(x^3-y^2) (x,y) [(x) -1 (Dx) 1 (y) -2 (Dy) 2]] wdeRham0
 Example 2 : [(x^3+y^3+z^3) (x,y,z)
      [(x) -1 (Dx) 1 (y) -2 (Dy) 2 (z) -3 (Dz) 3]] wdeRham0
 Example 3 : [(x^3 -y z^2) (x,y,z)
      [(x) -1 (Dx) 1 (y) -2 (Dy) 2 (z) -3 (Dz) 3]] wdeRham0
 Example 4 : [(x^3 -y^2 z^2) (x,y,z)
      [(x) -1 (Dx) 1 (y) -2 (Dy) 2 (z) -3 (Dz) 3]] wdeRham0
```