

Terence Etchells

```
i terates(vect, var, start, number)
Func
Local  inc, in, i terlist, numvar, tempvect
  If Left(string(var), 1)="/" Then
    Goto vector
  Else
    Goto scalar
  EndIf
Lbl scalar
{start}»i terlist
For inc, 1, number
  Limit(vect, var, start)»start
  augment(i terlist, {start})»i terlist
EndFor
Return listumat(i terlist)

Lbl vector
matulist(vect)»vect
matulist(var)»var
matulist(start)»start
dim(var)»numvar
start»i terlist
For inc, 1, number
  vect»tempvect
  For in, 1, numvar
    Limit(tempvect, var[in], start[in])»tempvect
  EndFor
  tempvect»start
  augment(i terlist, start)»i terlist
EndFor
Return listumat(i terlist, numvar)
EndFunc
```

```

iterate (vect, var, start, number)
Func
Local inc, in, iterlist, numvar, tempvect
If Left(string(var), 1) = "[" Then
    Goto vector
Else
    Goto scalar
EndIf
Lbl scalar
{start}»iterlist
For inc, 1, number
    Limit(vect, var, start)»start
    augment(iterlist, {start})»iterlist
EndFor
Return listumat(iterlist)

Lbl vector
matulist(vect)»vect
matulist(var)»var
matulist(start)»start
dim(var)»numvar
start»iterlist
For inc, 1, number
    vect»tempvect
    For in, 1, numvar
        Limit(tempvect, var[in], start[in])»tempvect
    EndFor
    tempvect»start
    augment(iterlist, start)»iterlist
EndFor
Return listumat(iterlist, numvar)
EndFunc

```

element(vect, number)

Func

Local vectdim

dim(vect)»vectdim

matulist(vect)»vect

If vectdim[1]=1 Then

Goto single

Else

Goto many

EndIf

Lbl single

Return vect[number]

Lbl many

Return listumat(mid(vect, (number-1)*vectdim[2]+1, vectdim[2]))

EndFunc

rev_element(vect)

Func

Local newlist, inc

matulist(vect)»vect

{ }»newlist

For inc, dim(vect), 1, ^1

augment(newlist, {vect[inc]})»newlist

EndFor

Return listumat(newlist)

EndFunc

Terence & Josef

```
swp_elem(vect, first, second)
Func
Local help
If first>second Then
    first»help: second»first: help»second
EndIf
If first=second Then
    Return vect
Else
    Local newlis, inc
    matulist(vect)»vect
    {}»newlis
    For inc, 1, first-1
        augment(newlis, {vect[inc]})»newlis
    EndFor
    augment(newlis, {vect[second]})»newli
    For inc, first+1, second-1
        augment(newlis, {vect[inc]})»newlis
    EndFor
    augment(newlis, {vect[first]})»newlis
    For inc, second+1, dim(vect)
        augment(newlis, {vect[inc]})»newlis
    EndFor
    Return listumat(newlis)
EndIf
EndFunc
```

del_elem(vect, elem)

```
Func
Local newl i s, i nc, si ze
{}»newl i s
di m(vect)»si ze
If si ze[1]=1 Then
  Goto vector
El se
  Goto matrix
EndIf
Lbl vector
matüli st(vect)»vect
For i nc, 1, el em-1
  augment(newl i s, {vect[i nc]})»newl i s
EndFor
For i nc, el em+1, di m(vect)
  augment(newl i s, {vect[i nc]})»newl i s
EndFor
Return l i stümat(newl i s)

Lbl matrix
matüli st(vect)»vect
For i nc, 1, si ze[2]*(el em-1)
  augment(newl i s, {vect[i nc]})»newl i s
EndFor
For i nc, si ze[2]*el em+1, di m(vect)
  augment(newl i s, {vect[i nc]})»newl i s
EndFor
Return l i stümat(newl i s, si ze[2])
EndFunc
```

denomi na(exp1)

```
Func
Local pl ace, si ze
factor(exp1)»exp1
stri ng(exp1)»exp1
di m(exp1)»si ze
i nStri ng(exp1, "/" )»pl ace
Return expr(ri ght(exp1, si ze-pl ace))
EndFunc
```

numerato(exp1)

```
Func
Local pl ace, si ze
factor(exp1)»exp1
stri ng(exp1)»exp1
di m(exp1)»si ze
i nStri ng(exp1, "/" )»pl ace
Return expr(l eft(exp1, pl ace-1))
EndFunc
```

lhs(eq)

Func

Return left(eq)

EndFunc

rhs(eq)

Func

Return right(eq)

EndFunc

Terence & Josef

```
plot_mat(mat)
Prgm
PlotsOff
Local xlist, ylist, reply, re1, re2, size
If dim(mat)[2] = 2 Then
  Disp "Must be a matrix of 2-D points"
Else
  matül ist(mat)»mat
  dim(mat)»size
  {}»xlist: {}»ylist
  For inc, 1, size, 2
    augment(xlist, {mat[inc]})»xlist
  EndFor
  For inc, 2, size, 2
    augment(ylist, {mat[inc]})»ylist
  EndFor
  "n"»reply
  Dialog
  Title "Autoscale"
  Request "Autoscale? (y/n)", reply
  EndDialog
  If reply="y" Then
    Local xrange, yrange, xma, xmi, yma, ymi, inc
    max(xlist)»xma: min(xlist)»xmi
    max(ylist)»yma: min(ylist)»ymi
    xma-xmi»xrange: yma-ymi»yrange
    xmi-.1*xrange»xmin
    xma+.1*xrange»xmax
    ymi-.1*yrange»ymin
    yma+.1*yrange»ymax
    ``xrange/10»xscl: yrange/10»yscl
    ZoomSqr
    Goto plot
  Else
    "n"»reply
    Dialog
    Title "Default Scale"
    Request "Default Scale? (y/n)", reply
    EndDialog
    If reply="y" Then
      ZoomStd
      ZoomSqr
      Goto plot
    Else
      Goto plot
    EndIf
  EndIf
EndIf
EndIf
EndIf

Lbl plot
```

```

Di alog
  Ti tle "Pl ot Opti ons"
  Request "Di sc. /Conn. ? (d/c)", re1
EndDI og
If re1 "c"
Goto di s
For i nc, 1, si ze-3, 2
  Li ne mat[i nc], mat[i nc+1], mat[i nc+2], mat[i nc+3]
EndFor
Goto end
Lbl di s
  Di alog
  Ti tle "Poi nt Si ze"
  Request "Smal l /Large? (s/l )", re2
EndDI og
If re2 "l " Then
  PtOn xli st, yli st
Else
  For i nc, 1, si ze/2, 1
    Ci rcle xli st[i nc], yli st[i nc], (xmax-xmi n)/120
  EndFor
EndIf
Lbl end
EndPrgm

```


W. Proepper

```
sol syst(sys, var)
Func
Local i, j, n, mtx, dtm
rowDim(sys)»n
newMat(n, n+1)»mtx
For i, 1, n
  For j, 1, n
    ¶(left(sys[i, 1]), var[j, 1])»mtx[i, j]
    right(sys[i, 1])»mtx[i, n+1]
  EndFor
EndFor
det(subMat(mtx, 1, 1, n, n))»dtm
If getType(dtm)="NUM" and dtm 0 or getType(dtm)="EXPR" Then
  rref(mtx)»mtx
  Return subMat(mtx, 1, n+1, n, n+1)
Else
  Return "Ni cht ei ndeuti g l ösbar"
EndIf
EndFunc
```

F1

F2

F3

F4

F5

F6

Algebra

Calc

Other

PrgmIO

Clear a-z...

iterates($\sqrt{x+1}, x, \sqrt{2}, 3$)

$\sqrt{2}$
 $\sqrt{\sqrt{2}+1}$
 $\sqrt{\sqrt{\sqrt{2}+1}+1}$
 $\sqrt{\sqrt{\sqrt{\sqrt{2}+1}+1}+1}$

iterate($\sqrt{x+1}, x, \sqrt{2}, 4$)

$\sqrt{\sqrt{\sqrt{\sqrt{\sqrt{2}+1}+1}+1}+1}$

[a b 1 2 3]→v0

[a b 1 2 3]

element(v0,4)

2

element(v0,4)

TIPRG

DEG EXACT

SEQ 4/10

F1

F2

F3

F4

F5

F6

Algebra

Calc

Other

PrgmIO

Clear a-z...

[a b 1 2 3]→v0

[a b 1 2 3]

element(v0,4)

2

rev_elem(v0)

[3 2 1 b a]

del_elem(v0,4)

[a b 1 3]

denomina($\frac{a+b}{a \cdot (x^2+y^2)}$)

$a \cdot (x^2+y^2)$

numerato($\frac{a+b}{a \cdot (x^2+y^2)}$)

a+b

numerato((a+b)/(a*(x^2+y^2)))

TIPRG

DEG EXACT

SEQ 6/6

F1

F2

F3

F4

F5

F6

F7

Zoom

Trace

ReGraph

Math

Draw

2 2

-2 2

-2 -2

2 -2

2 2

plot_mat(squ)

Done

TIPRG

DEG EXACT

SEQ

