```
!MS3-RC-Workout.f95
!2025.01.25.1440cst JMS- Sequence->Emulator->Results
!                     - Generates a lookup table
!              Computer- "T4"/HP-800-G4-Mini/i7-8700T/IntelUHD630/Win10Pro-22H2
!                        ^name  ^Mfgr.Id    ^chipset ^graphics   ^OS
!                           /AbsoftProFortran 21.0.2/OpenGL+Glut3.6
!                            ^compiler              ^Fortran graphics
!
!Table of Contents - ToC:
!Subroutine MinStepSolver(iP)                       -Lookup tables generator
!                                                   -opens many files
!Subroutine BitLog(ModeF1R2,n8,nNewout,iP)          -Results (huge) indexer
!Subroutine ReadYourNml(EnvNml,Ur,Us,Ut,cOutFile)   -opens 'MS3.ini' & EnvNml
!                                          ...called by MS1Launch at the outset.
!Subroutine ImportEin(iP)                           -internalizes .nml data
!                                          ...and branches to apps via %iType.
!Subroutine PrintErec(Ein,iP,Label)                 -view Emulator setup
!Subroutine PrintSRrec(SortL,iP,Label)              -view a Sequence-or-Result
!-------------------------------------------------------------------------------7 9

Subroutine MinStepSolver(iP)
!2025.01.19.0925cst JMS- the Minimum Step Solver for Rubik's Cubes,
!                        exercising: EmulatorRC() & IndexerRC()
!--Globals
 use MS1Def   ,only: Ur,Us,Ut,Uread1,Uwrite1,Uwrite2,Urdwr,RunSecs
 use MS2RCDef ,only: cTitle,RubSize,iType,cFileOut,cLuTOut,EnvNml  & !housekeep
                    ,Mmax,Ctot,C,nS8,Vtot,V,Vtotshow,VA,cVAtest    & !Sequences
                    ,H8 ,nRaccum8,Rtot8,SRrec,resetSR,Srseed,RV    & !& Results
                    ,Sr1,Srn,Srbase,Srtry,Stried8,Sfound8,RVbase   &
                    ,nSeqs,nSeqPrev,nSeqPres                       &
                    ,Ein                                           & !inputs
                    ,Csymbol,Asymbol                                 !symbology

!use RuBbMod !,only: _______    !:use/modify module variables    JMS 2023.07.14
!--End Globals
 implicit none
 !--Arguments
  integer(4)::iP          !Write enable>5: write(iP,...)
 !--Internals
 type(SRrec)::Srtest      !Ein is transcribed to SrTest
  integer(4)::Init
  integer(4)::i,iii,n
 !integer(4)::iAlloc      !Array allocation flag =0:success
  integer(4)::nSe
  character(22)::Datime22L
  integer(4)::nin         !Recursion Level index- `called by level
  integer(8)::H8a,Indexer
  integer(4)::nNew        !=1:new, =-1:not new
  integer(8)::nMprev
  integer(4)::nC
  integer(4)::valid       !=0:invalid; =nCin:valid
  integer(8)::n8,n8fnd,nS8H8 ,nBeam8
  integer(4)::Vtotl
  character(42)::cLabel42

  real(8)   ::rRate1,rRate2,rTime1,rTime2
  real(8)   ::RunSecs01,RunSecs10,RSecs0010(2,-1:40)
  integer(1)::Zero1  = 0
  integer(1)::Minus1 = -1
  integer(4)::nZ
  !--EndDefs-------------------------------------------------

```

```
63      if(Init==0) then
64
65        write(Us,"('ReadYourNml(): reading ',:,79a1)") &
66                                  (EnvNml(i:i),i=1,len_trim(EnvNml))
67        do i = 1,8; Ein%cVAtest(i:i) = char(Asymbol(Ein%VA(i))); enddo!i
68          call PrintErec(Ein,Us,'MinStepSolver: Ein% inputs')
69        if(iP>5) then
70          call PrintErec(Ein,iP,'MinStepSolver: Ein% inputs')
71
72         !--Internalize the variables in the Ein record:
73          cTitle    = Ein%cTitle
74          cFileOut  = Ein%cSummaryOut
75          cLuTOut   = Ein%cRrAsciiOut
76          iType     = Ein%iType
77          RubSize   = Ein%RubSize
78          Vtot      = Ein%Vtot      !Voters - total
79          V         = 0
80          cVAtest   = Ein%cVAtest
81          V(1:Vtot) = Ein%V(1:Vtot) !Voter(s) | cells          in use
82          Ctot      = Ein%Ctot      !Choices- total
83          C         = 0
84          C(1:Ctot) = Ein%C(1:Ctot) !Choices  | face rotations in use
85          Mmax      = Ein%Mmax
86          Vtotshow  = Ein%Vtotshow
87            !--Define Srtest
88          Srtest = resetSR  !clears the Srtest record
89          Srtest%nS8       =       0_8
90          Srtest%H8        =       0_8 !undefined
91
92          Srtest%Mused    = Ein%Mtotvis
93          Srtest%MC(1:30) = Ein%MCvis(1:30)
94          Srtest%Vtot     = Ein%Vtot
95                 Vtotl    = Ein%Vtot
96          Srtest%cVAsymb(1:VtotL)  = Ein%cVAtest(1:VtotL)
97          Srtest%VA(     1:VtotL)  = Ein%VA(     1:VtotL)
98
99         !--Zero the Sequences and Results counters;
100         H8  = 0_8
101       endif!(iP>5)
102       call SaveOutFile
103       Init = 1
104     endif!(Init==0)
105
106    !Seed the Min-Steps Generator:
107              nSeqPrev  =    0
108     nSeqs(1:4,nSeqPrev) = (/ 0, 1, 1, 1 /) !(nSeq|n1st|nLast|count,Sequence#)
109
110       Srseed                 = resetSr
111       Srseed%nS8             = 1_8
112       Srseed%Vtot            = Vtot
113       Srseed%VA              =-1
114       Srseed%cVAsymb         = ' '
115     do i = 1,Vtot
116       Srseed%VA(i)           = 1
117       Srseed%cVAsymb(i:i)    = char(Asymbol(1))  !='a'
118     enddo!i
119       Srseed%Mused           = 0
120       Srseed%MC(1:30)        = (/ 0, (-1,iii=1,29) /) !-1:undefined
121              Stried8         = 1_8
122              Sfound8         = 1_8
123       Srseed%H8              = 1_8
124
125       Srseed%MC(2:(Srseed%Vtot+1)) = V(1:Vtot)
```

```
125       Srseed%MC(2:(Srseed%Vtot+1)) = V(1:Vtot)
126       Srseed%MC(   29            ) = Ctot
127
128                   nin = 0
129     H8a  = Indexer(nin,Srseed,iP) !,iP)  <- print Indexer()'s setup here
130
131     if(iP>5) then
132       write(iP,"(' n, RVbase(n)%nAloc(1:24):')")
133       write(iP,"( 5x,'a  b  c  d  e  f  g  h  i  j  k  l  m  n  o  p  q  r',\)")
134       write(iP,"(  '  s  t  u  v  w  x')")
135       do n = 1,Ein%Vtot
136         write(iP,"(i2,': ',i2, 23(',',i2) )")  &
137                           n,   (RVbase(n)%nAloc(i),i=1,24)
138       enddo!n
139       write(iP,*)
140     endif!(iP>5)
141
142                               Srseed%H8    = H8    != H8a too
143                                    Rtot8 = nRaccum8
144     if(iP>5) call PrintSRrec(Srseed,iP,'MinStepSolver: Srseed% results @L146:')
145     if(iP>6) call SaveOutFile
146
147                                nNew = 0   !------------------
148    !call BitLog(ModeF1R2,n8         ,nNew,iP) !F1R2:1=Find,2=Report,3=dealloc.
149     call BitLog(       1,Srseed%H8 ,nNew, 0)
150     if(iP>5) write(iP,*)
151     call PrintSRrec(     Srseed,iP,'MinStepSolver @L153: Srseed%')
152     nS8 = 1_8
153                   nin = 0
154     H8a  = Indexer(nin,Srseed, 0)   ! |||||||||||||||||||||
155     call SaveOutFile
156
157    !--Set up Sr() as a direct access file:      2024.08.22
158      !To screen:
159       write( 6,"(/'Set up Sr as a direct access file:  Unit=',i2)") Urdwr
160       call nRunSec(-10,RunSecs10, 6)
161       write( 6,"('nRaccum8         =',i25         )") nRaccum8
162       write( 6,"('c_sizeof(Srtry)  =',i25         )")          c_sizeof(Srtry)
163       write( 6,"('Sr.bim filesize <=',i26         )") nRaccum8*c_sizeof(Srtry)
164     if(iP>6) then
165      !To file:
166       write(iP,"(/'Set up Sr as a direct access file:  Unit=',i2)") Urdwr
167       call nRunSec(-10,RunSecs10,iP)
168       write(iP,"('nRaccum8         =',i25         )") nRaccum8
169       write(iP,"('c_sizeof(Srtry)  =',i25         )")          c_sizeof(Srtry)
170       write(iP,"('Sr.bim filesize <=',i25         )") nRaccum8*c_sizeof(Srtry)
171       if(nRaccum8*c_sizeof(Srtry)<0) then
172         write(iP,"(/'Sr.bim filesize <0: is unopenable, @L174, Will halt.')")
173         call SaveOutFile
174       endif!(nRaccum8*c_sizeof(Srtry)<0)
175     endif!(iP>6)
176     if(nRaccum8*c_sizeof(Srtry)<0) stop 'Sr.bim is unopenable when <0, @L178.'
177
178     open(Urdwr, file=Ein%cSrBinary, action='readwrite' &   !-> 88 bytes/record
179                , access='direct'    , recl  = C_SIZEOF(Srtry), status='replace')
180
181     Sr1       = Srseed
182     write(Urdwr,rec = 1_8) Sr1
183
184     Srbase    = resetSr
185     Srn       = resetSr
186
```

```
187   !Initialize the clocks:
188    RSecs0010 = 0.d0
189    call nRunSec( -1,RunSecs01, 0); RSecs0010(1,-1) = RunSecs01
190    rRate1 = 30000.d0
191    rRate2 = 30000.d0
192
193    if(iP>5) write(iP, &
194     "(/17x,'***** MinStepSolver() @L196: commencing loop... *****'/)") !~~~~~~
195
196   !--Top of the Min-Step Solver loop:
197  10 continue
198    nSeqPres = nSeqPrev+1
199   !Update the clocks
200           call nRunSec( +1,RunSecs01, 0)
201    RSecs0010(1,nSeqPrev) = RunSecs01
202    RSecs0010(2,nSeqPrev) = RSecs0010(1,nSeqPrev)-RSecs0010(1,nSeqPrev-1)
203
204    if(nSeqPrev>1) then
205     !Update the rate & time approximations:
206     !Cumulative performance:
207      if((     nSeqs(3,nSeqPrev  )>0_8).and.(   Rsecs0010(1,nSeqPrev)>0.d0)) &
208      rRate1 = nSeqs(3,nSeqPrev  )       /      Rsecs0010(1,nSeqPrev)
209      rTime1 = nRaccum8                  /      rRate1
210     !nSeq = _ performance:
211      if((     nSeqs(4,nSeqPrev-1)>0_8).and.(   Rsecs0010(2,nSeqPrev)>0.d0)) &
212      rRate2 = nSeqs(4,nSeqPrev-1)       /      Rsecs0010(2,nSeqPrev)
213      rTime2 = nSeqs(4,nSeqPrev  )       /      rRate2
214
215   !--Output to screen:
216      write( 6, &
217     "(/' nSeq,      n1st,      nLast,      count',14x,'Level    <sec>      Run')")
218      do nSe = 0,nSeqPrev
219                          write(6,"(i4,3i12,2f19.3)")      &
220                           nSeqs(1:4,nSe),Rsecs0010(2,nSe) &
221                                         ,Rsecs0010(1,nSe) &
222                                       - Rsecs0010(1,  0)
223      enddo!nSe
224      write( 6,"(/i4,6x,i18,f16.3,'/sec      est.=',f19.3)")  &
225           nSeqPrev+1, nSeqs(4,nSeqPrev),rRate2,rTime2
226      write( 6,"( 7x,'if real:',i13,f16.3,'/sec      est.=',f19.3)")  &
227                         nRaccum8         ,rRate1,rTime1
228        call jdate22(DaTime22L)
229        write( 6,"(56x,a22)") DaTime22L
230
231      if(iP>6) then
232       !--Output to file:
233        write(iP, &
234     "(/'nSeq,       n1st,      nLast,      count',14x,'Level    <sec>     Run')")
235        do nSe = 0,nSeqPrev
236                             write(iP,"(i4,3i12,2f19.3)")     &
237                              nSeqs(1:4,nSe),Rsecs0010(2,nSe) &
238                                            ,Rsecs0010(1,nSe) &
239                                          - Rsecs0010(1,  0)
240        enddo!nSe
241        write(iP,"(/i4,6x,i18,f16.3,'/sec      est.=',f19.3)")  &
242             nSeqPrev+1, nSeqs(4,nSeqPrev),rRate2,rTime2
243        write(iP,"( 7x,'if real:',i13,f16.3,'/sec      est.=',f19.3)")  &
244                           nRaccum8,rRate1,                 rTime1
245        write(iP,"(56x,a22)") DaTime22L
246        call SaveOutFile
247       endif!(iP>6)
248     endif!(nSeqPrev>1)
```

```
249
250     if(nSeqPres>Ein%Mmax) then
251       if(iP>6) write(iP,"(/'@L253: Exceeded Ein%Mmax. Reporting results...')")
252       goto 90
253     endif!(nSeqPres>Ein%Mmax)
254
255     if(nSeqs(4,nSeqPrev)== 0) goto 90
256
257    !--Time the next level
258     call nRunSec(-10,RunSecs10, 0)
259     call jdate22(DaTime22L)
260
261    !--The outer loop of the core process:
262    !  Testing each previously valid Sr()'s:
263     do nMprev = nSeqs(2,nSeqPrev),nSeqs(3,nSeqPrev)
264
265      !if(nSeqPrev>6) then
266       if(nSeqs(4,nSeqPrev)>80) then
267                       nBeam8 = nMprev+1_8-nSeqs(2,nSeqPrev)
268         if(mod(       nBeam8,nSeqs(4,nSeqPrev)/80)==1_8) &
269         call Beamer(nBeam8,nSeqs(4,nSeqPrev))
270       endif!(nSeqPrev>5)
271       read(Urdwr,rec = nMprev) Srbase
272
273      !--The inner loop of the core process:
274      !  Evaluate SrBase using all possible move Choices:
275       do nC = 1,Ctot
276        !--Excercise the system:
277         call EmulatorRC(Srbase,nC,Srtry,valid, 0) !iP)
278         Stried8 = Stried8 +1_8
279                       nin = 0
280        !--Compute H8  (the index#) for the resulting set of Attitudes:
281         H8a  = Indexer(nin,Srtry, 0)
282
283        !--Determine if Srtry%H8  is novel or known:
284         call BitLog(     1  ,Srtry%H8 ,nNew, 0) !,iP)
285
286        !--If Srtry is new- append it to the list of Sr()'s
287         if(nNew==1) then
288                 Sfound8  = Sfound8+1_8
289           Srtry%nS8      = Sfound8
290          !Sr(   Sfound8) = Srtry
291           write(Urdwr,rec = Sfound8) Srtry
292           call PrintSRrec(Srtry, 0,'MinStepSolver@L294: saved:')
293          else
294           call PrintSRrec(SRtry, 0,'MinStepSolver@L296: discarded:')
295         endif!(nNew==1)
296       enddo!nC
297     enddo!nMprev
298
299    !--Update the sequence table:
300     call SaveOutFile
301     if(Sfound8>nSeqs(3,nSeqPrev)) then
302       nSeqs(1,nSeqPres) = nSeqPres
303       nSeqs(2,nSeqPres) = nSeqs(3,nSeqPrev)+1
304       nSeqs(3,nSeqPres) = Sfound8
305       nSeqs(4,nSeqPres) = Sfound8-nSeqs(3,nSeqPrev)
306       nSeqPrev = nSeqPres
307       goto 10     !This is the bottom of the MinStepSolver loop.
308     endif!(nSortsTot>nSeqs(3,nSeqPrev))
309
310     write( 6,"(/'@L312: Novel Results exhausted; drop through.')")
```

```
311
312  !-----------------------------------
313
314  !--The search for novel Sr()'s is done, and Sr()%H8  values are defined.
315 90 continue
316
317    write( 6,"('90 @L320:: will deallocate BitLog...')")
318    call BitLog(       3,1000002_8,nNew,iP)  !Close BitLog
319    close(Urdwr) !This saves all of Ein%cSrBinary to disk
320
321   !--Complete Ein Zone update and export Ein as a binary file:
322    Ein%Einsize           = sizeof(Ein)
323    Ein%Zonebimsize       = nRaccum8
324    Ein%ZoneFound         = Sfound8
325        nZ                = Ein%nZ
326    if( nZ>0) then
327      Ein%cZrFilename(nZ) = Ein%cRrtoDisBinary
328      Ein%Vsizez          = 0
329      do n=1,VtotL
330        Ein%Vsizez(n)     = RV(n)%nAavail
331      enddo!n
332    endif!nZ>0
333    Ein%Mtot              = nSeqPrev
334
335    call PrintErec(Ein,iP,'MinStepSolver @L340: Ein% output')
336
337    write( 6,"('Export the -Ein.bim file:')")
338    if(iP>6) &
339    write(iP,"('Export the -Ein.bim file:')")
340    open(Uwrite1, file=Ein%cEinBinary, action='write'              &
341                , access='direct'       , recl = sizeof(Ein)    , status='replace' )
342        write(Uwrite1,rec=1 ) Ein
343    close(Uwrite1)
344    if(iP>6) then
345      write(iP,"('Population of -Ein.bim:    completed.')")
346      call nRunSec(+10,RunSecs10,iP)
347      call nRunSec(  0,RunSecs  ,iP)
348      call SaveOutFile
349    endif!(iP>6)
350
351    open(Urdwr, file=Ein%cSrBinary, action='read' &   !-> 88 bytes/record
352                , access='direct'    , recl  = C_SIZEOF(Srtry), status='old')
353   !--Export the results in various ways
354    write(6,"(/'The first three Sr() records are:')")
355    do n8 = 1_8,3_8
356      write(cLabel42,"('MinStepSolver @L362: Sr(',i1,'):')") n8
357      read(Urdwr,rec = n8) Srn
358      call PrintSRrec(Srn, 6,cLabel42) ;write(6,*)
359      if(iP>6) then
360        call PrintSRrec(Srn,iP,cLabel42)
361        call SaveOutFile
362        write(iP,*)
363      endif!(iP>6)
364    enddo!n8
365    write(6,"(/'The last three Sr() records are:')")
366    do n8 = Sfound8-2_8,Sfound8
367      write(cLabel42,"('MinStepSolver @L373: Sr(',i15,'):')") n8
368      read(Urdwr,rec = n8,err=91) Srn
369      call PrintSRrec(Srn, 6,cLabel42) ;write(6,*)
370      if(iP>6) then
371        call PrintSRrec(Srn,iP,cLabel42)
372        call SaveOutFile
```

```
373         write(iP,*)
374       endif!(iP>6)
375     enddo!n8
376
377     goto 92                                                 !cLabel42 was too short.
378  91 pause 'Error MS3-RC-Workout.f95@L384: Exporting the last 3 records.'
379  92 continue
380
381    !--Export the Rr Binary Cross Reference to Sr:
382     if(iP>5) then
383       write(iP,"(/'Export the -RrtoSr.bim file:')")
384       call nRunSec(-10,RunSecs10,iP)
385       write(iP,"('nRaccum8  (clear)     =',i12)") nRaccum8
386       write(iP,"('    if @78654.469/sec =',f16.3,' sec')") nRaccum8/78654.469_8
387       write(iP,"('Sfound8   (write)     =',i12)") Sfound8
388       write(iP,"('    if @72208.006/sec =',f16.3,' sec')") Sfound8/72208.006_8
389       write(iP,"(' ^ sum                =',f16.3,' sec')") &
390                                       nRaccum8/78654.469_8+Sfound8/72208.006_8
391       write(iP,"('Rr-to-Sr.bim filesize =',i12,' bytes')") nRaccum8*8_8
392       call SaveOutFile
393     endif!(iP>5)
394       write( 6,"(/'Export the -RrtoSr.bim file:')")
395       call nRunSec(-10,RunSecs10, 6)
396       write( 6,"('nRaccum8  (clear)     =',i12)") nRaccum8
397       write( 6,"('    if @78654.469/sec =',f16.3,' sec')") nRaccum8/78654.469_8
398       write( 6,"('Sfound8   (write)     =',i12)") Sfound8
399       write( 6,"('    if @72208.006/sec =',f16.3,' sec')") Sfound8/72208.006_8
400       write( 6,"(' ^ sum                =',f16.3,' sec')") &
401                                       nRaccum8/78654.469_8+Sfound8/72208.006_8
402
403    !Open & initialize the file:
404     write( 6,"('Clear the -RrtoSr.bim file:')")
405     if(iP>6) &
406     write(iP,"('Clear the -RrtoSr.bim file:')")
407     open(Uwrite1, file=Ein%cRrtoSrBinary, action='write'                  &
408                 , access='direct'       , recl = 8      , status='replace' )
409           ! write(12,rec= 1) 1
410       do n8 = 1,nRaccum8
411         if(mod(n8,5000_8)==1_8) call Beamer(n8,nRaccum8)
412         write(Uwrite1,rec=n8) 0_8
413       enddo!n8
414       write(6,*)
415     close(Uwrite1)
416             call nRunSec(+10,RunSecs10,06)
417             call nRunSec(  0,RunSecs  ,06)
418     if(iP>5 )write(iP,"('Initializing Rr-to-Sr.bim:     completed')")
419     if(iP>5) call nRunSec(+10,RunSecs10,iP)
420     call SaveOutFile
421
422    !Reopen & write the file:
423     write( 6,"('Populate the -RrtoSr.bim file:')")
424     if(iP>6) &
425     write(iP,"('Populate the -RrtoSr.bim file:')")
426     open(Uwrite1, file=Ein%cRrtoSrBinary, action='readwrite'              &
427                 , access='direct'       , recl = 8          , status='old' )
428       do n8 = 1,Sfound8
429         if(mod(n8,5000_8)==1_8) call Beamer(n8,Sfound8)
430         read(Urdwr,rec = n8) Srn
431         write(Uwrite1,rec=Srn%H8 ) Srn%nS8
432       enddo!n8
433       write(6,*)
434     close(Uwrite1)
```

```
435     if(iP>6) then
436       write(iP,"('Population of Rr-to-Sr.bim:    completed.')")
437       call nRunSec(+10,RunSecs10,iP)
438       call nRunSec(  0,RunSecs  ,iP)
439       call SaveOutFile
440     endif!(iP>6)
441     call nRunSec(+10,RunSecs10, 6)
442     call nRunSec(  0,RunSecs  , 6)
443
444   !--Export Sr as Text :
445       write( 6, &
446        "(/'Export Rr(1:Sfound8) (founds only) as -RrAscii.txt (<= 4.e6)')")
447       write( 6,"('& Export the -RrToDis.bim file:')")
448       write( 6,"('-RrToDis.bim filesize  =',i12,' bytes')") nRaccum8
449       call nRunSec(-10,RunSecs10, 6)
450     if(iP>6) then
451       write(iP, &
452        "(/'Export Rr(1:Sfound8) (founds only) as -RrAscii.txt (<= 4.e6)')")
453       write(iP,"('& Export the -RrToDis.bim file:')")
454       write(iP,"('-RrToDis.bim filesize  =',i12,' bytes')") nRaccum8
455       call nRunSec(-10,RunSecs10,iP)
456       call SaveOutFile
457     endif!(iP>6)
458
459   !--Export the -RrtoSr.bim & -RrtoDis.bim file:
460
461     open(Uread1   , file=Ein%cRrtoSrBinary , action='read'   &
462                   , access='direct'        , recl = 8      , status='old'  )
463
464       open(UWrite1, file=Ein%cRrAsciiOut   , action='write', status='replace')
465
466       open(UWrite2, file=Ein%cRrtoDisBinary, action='write'  &
467                   , access= 'direct'       , recl = 1      , status='replace')
468
469                   n8fnd = 0_8
470        !Write the file header:
471         write(UWrite1,"(:,79a1)") &
472               (Ein%cRrAsciiOut(i:i),i=1,len_trim(Ein%cRrAsciiOut))
473         write(UWrite1,"(2i3,'    ',\)") &
474             RubSize,Vtot
475         do i = 1,Vtot
476           write(UWrite1,"(i3,\)") V(i)
477         enddo!i
478         write(UWrite1,"('  RubSize, Vtot, V(1:Vtot)')")
479         write(UWrite1,"(i11,i21,i3,'  Rtot,Stot,Mtot')") &
480                       nRaccum8,Sfound8,nSeqPrev
481        !-----------------------------
482             n8fnd = 0_8
483         do n8 = 1,nRaccum8
484           if(mod(n8,5000_8)==1_8) call Beamer(n8,nRaccum8)
485           read(Uread1,rec=n8) nS8H8
486           if(nS8H8 ==0_8) then
487             write(Uwrite2,rec = n8        ) Minus1 ! -RrToDis.bim
488            else                                    !   = -1 implies undefined -&-
489             n8fnd = n8fnd +1_8                     !   =  0 implies solved
490             read( Urdwr  ,rec = nS8H8 ) Srn
491
492             if(n8<10_8) then
493               write(cLabel42,"('MinStepSolver @L502: Sr(',i10,'):')") nS8H8
494               call PrintSRrec(Srn,iP,cLabel42)
495               write(iP,*)
496             endif!(n8<10_8)
```

```
497
498             write(Uwrite2,rec = n8 ) Srn%Mused !-RrToDis.bim  2024.11.21
499
500             if(n8fnd>4000000_8) cycle  !Supports 2by:3674160
501             write(UWrite1,"(i11,1x,a12,1x,i11,1x,i2,1x,:,22(1x,a1))") &
502                              Srn%H8                                    &
503                             ,Srn%cVAsymb                 & ! Up to 12 print out.
504                             ,Srn%nS8                     &
505                             ,Srn%Mused                   &
506                     ,(Csymbol(Srn%MC(i)),i=1,Srn%Mused)    ! first 20 fit inside
507           endif!(nS8H8 ==0_8)                              ! 80-column printout.
508         enddo!n8
509         call SaveOutFile
510
511         write(6,*)
512        !-----------------------------
513         write(Uwrite1, &
514     "(/'nSeq,      n1st,     nLast,     count',15x,'Level    <sec>       Run')")
515         do nSe = 0,nSeqPrev
516           write(Uwrite1,"(i4,3i11,2f20.3)")      &
517           nSeqs(1:4,nSe),  Rsecs0010(2,nSe) &
518                          ,  Rsecs0010(1,nSe) &
519                           - Rsecs0010(1,  0)
520         enddo!nSe
521         Write(Uwrite1, &
522         "('     Allocated:',i11,'   ...Rr()`s address space size')") nRaccum8
523         write(Uwrite1, &
524         "(/'Export Rr(1:Sfound8) (founds only), in found order, completed.')")
525         call nRunSec(+10,RunSecs10,UWrite1)
526         call nRunSec(  0,RunSecs  ,UWrite1)
527         call jdate22(DaTime22L)
528         write(UWrite1,"(a22)") DaTime22L
529       close(Uwrite1) !-RrASCII.txt
530       close(Uwrite2) !-RrToDis.bim
531     close(Uread1)    !-RrtoSr.bim
532
533     close(Urdwr)     !-Sr.bim
534
535       write( 6,"('Export Rr(1:Sfound8) & RrToDis completed.  iP =',i3)") iP
536       call nRunSec(+10,RunSecs10, 6)
537       call nRunSec(  0,RunSecs  , 6)
538     if(iP>6) then
539       write(iP,"('Export Rr(1:Sfound8) & RrToDis completed.  iP =',i3)") iP
540       call nRunSec(+10,RunSecs10,iP)
541       call nRunSec(  0,RunSecs  ,iP)
542       call SaveOutFile
543     endif!(IP>6)
544                                                                        return
545 End Subroutine MinStepSolver
546 !-------------------------------------------------------------------------7 9
547
548 Subroutine BitLog(ModeF1R2,n8,nNewout,iP)
549 !2024.08.22.0920cdt JMS- Using bits to track "used`s". <= 30 billion
550 !--Globals
551 !--End Globals
552  implicit none
553  !--Arguments
554   integer(4)  ::ModeF1R2 !=1:fill, =2:read; Any other values deallocate & reset
555   integer(8)  ::n8       !number to evaluate
556   integer(4)  ::nNewout  !=1:new, =-1:not new
557   integer(4)  ::iP       !Write enable>5: write(iP,...)
558  !--Internals
```

```
559   integer(8),allocatable  ::nArray8(:) !( 500000000_8)  !use 60 of the <-63 bits
560                       !=   500,000,000  =   ~4 billion bytes (~8 bits/byte)
561                       !                     &  60 bytes per integer(8)
562                       != 30,000,000,000  = 30 billion indices
563   integer(8)  ::n8max = 30000000000_8
564                       !=
565                       != 30,000,000,000 != 30 billion indices
566   integer(4)  ::Init,nNew
567   integer(8)  ::i8,j8,Bit8,nSum8
568   !--EndDefs-------------------------------------------------
569    if(Init==0) then
570      allocate(nArray8(500000000_8))
571         !allocation avoids an 80,000,000 bytes compiler limit.
572      nArray8 = 0_8
573      nSum8   = 0_8
574      Init    = 1
575    endif!(Init==0)
576
577    if((n8<1).or.(n8>n8max)) then
578      call SaveOutFile
579      if(n8<   1_8) pause 'BitLog @L588: n8< 1, will halt.'
580      if(n8<   1_8) stop
581      if(n8>=n8max) pause 'BitLog @L590: n8>=n8max, will halt.'
582      if(n8>=n8max) stop
583    endif!((n8<1).or.(nL>n8max))
584
585   !--Check for previous use:
586    i8   = ( n8 - 1_8 ) / 60_8 + 1_8
587    Bit8 = n8 - ( i8 - 1_8 ) * 60_8
588    j8   = nArray8(i8)
589                        nNew    =  1
590    if(Btest(j8,Bit8)) nNew    = -1
591                        nNewout = nNew
592    select case(ModeF1R2)
593     case(1) !Fill
594      if(nNew==1) then; j8 = iBset(j8,Bit8)
595        nArray8(i8) =   j8
596        nSum8 = nSum8+1_8
597        if(iP<6)                                                          return
598        if(nSum8==1_8) &
599        write(iP,"(/'BitLog:                      Count   nNew')")
600        write(iP,"('n8=',i12,' --> ',i12,i6,'  bit set')") n8,nSum8,nNew
601       else
602        if(iP>5) &
603        write(iP,"('n8=',i12,' old',13x,i6)") n8,nNew
604      endif!(nNew==1)
605     case(2) !Read
606      if(iP<6)                                                          return
607      if(nNew==1) then
608        write(iP,"('n8=',i12,' free',12x,i6)") n8,nNew
609       else
610        write(iP,"('n8=',i12,' used',12x,i6)") n8,nNew
611      endif!(nNew==1)
612     case default !Deallocate, reset, and un-initialize:
613      if(iP>5) write(iP,"( 'BitLog: nSum8     = ',i12,'  clearing...')") nSum8
614      if(allocated(nArray8)) then
615        if(iP>5) write(iP,"('         deallocating nArray8.')")
616        deallocate(narray8)
617      endif!(allocated(nArray8))
618      nSum8   = 0_8
619      Init    = 0
620    end select!(ModeF1R2)
```

```
621                                                                     return
622 End Subroutine BitLog
623 !Example usage: 2024.07.24.1636
624 !    call BitLog(1,1000001_8,nNew,iP)   !<-Find
625 !    call BitLog(1,1000001_8,nNew,iP)   !<-Find
626 !    call BitLog(2,1000001_8,nNew,iP)   !<-Report
627 !    call BitLog(2,1000002_8,nNew,iP)   !<-Report
628 !    call BitLog(3,1000002_8,nNew,iP)   !<-Deallocate nArray8(:)
629 !--------
630 !BitLog:                     Count
631 !n8=     1000001 -->            1
632 !n8=     1000001 old
633 !n8=     1000001 used
634 !n8=     1000002 free
635 !BitLog: nSum8     =            1     Reset:
636 !       deallocating nArray8.
637 !-------------------------------------------------------------------------7 9
638
639 Subroutine ReadYourNml(EnvNml,Ur,Us,Ut,cOutFile)
640 !2025.01.19.0835cdt JMS- Reading "MS3.ini" to identify & read  *.nml
641 !--Globals
642  use MS2RCDef, only: Ein
643 !--End Globals
644  implicit none
645  !--Arguments          !These variables are in MS1Def:
646   character*79::EnvNml !  but are not "used" herein.
647   integer(4)  ::Ur
648   integer(4)  ::Us
649   integer(4)  ::Ut
650   character*79::cOutFile
651  !--Internals
652   integer(4)::i
653   logical   ::LExists         !File existence flag
654   integer(4)::Utin
655
656  NAMELIST / NnNml /  Utin,Ein
657
658    !--EndDefs-----------------------------------------------
659 !  Modifying key variables at runtime using NameList file `Sn.nml`:
660 !  Initial defaults are set here:
661    EnvNml   = 'MS-RC-R3-Scrambler.nml' !which specifies "NmNml"
662    Ut       =   6                      !Output unit#: Set to 6 or 13
663    cOutFile = '.\R3-Scrambler-Summary.txt'
664
665    write(Us,"('ReadYourNml(): reading `MS3.ini` file to ID the *.nml file')")
666    inquire(file = 'MS3.ini', exist = LExists)
667    if(LExists)then
668      open( unit=Ur, file  ='MS3.ini'    , action='read'      &
669                   , access='sequential', status='old' ,err=4 )
670        read(Ur,*,err=4) EnvNml
671      close(Ur); goto 5
672 4    pause "Error reading `MS3.ini` file. Press enter to continue."
673      close(Ur)
674     else
675      write(Us,"('ReadYourNml(): Didn`t find `MS3.ini` @L680.')")
676    endif!LExists
677 5  continue
678
679   write(uS,*) EnvNml
680
681   !Sn.nml Ein record listing: SER-Rubik-Def.f95 @L690+.
682    write(Us,"('ReadYourNml(): reading ',:,79a1)") &
```

```
683                             (EnvNml(i:i),i=1,len_trim(EnvNml))
684     inquire(file=EnvNml,exist=LExists)
685     if(LExists)then
686       open( unit=Ur, file=EnvNml         , action='read'    &
687                    , access='sequential', status='old' ,err=9)
688         read(Ur,nml=NnNml,err=8)         !<--Defined above
689       close(Ur); goto 10
690 8     pause "Error reading  NnNml @L699. Press enter to continue."
691       close(Ur); goto 10
692 9     pause "Error opening `NnNml.nml` file @L701. Press enter to continue."
693       close(Ur)
694      else
695       write(Us,"('ReadYourNml(): Didn`t find `EnvNml` @L704.')")
696     endif!LExists
697 10  continue
698     Ein%cSummaryOut   = Ein%NameRoot(1:len_trim(Ein%NameRoot))//"-Summary.txt"
699
700     Ein%cRrAsciiOut   =Ein%NameRoot(1:len_trim(Ein%NameRoot))//"-RrAscii.txt"
701     Ein%cSrBinary     =Ein%NameRoot(1:len_trim(Ein%NameRoot))//"-Sr.bim"
702     Ein%cRrtoSrBinary =Ein%NameRoot(1:len_trim(Ein%NameRoot))//"-RrtoSr.bim"
703     Ein%cEinBinary    =Ein%NameRoot(1:len_trim(Ein%NameRoot))//"-Ein.bim"
704     Ein%cRrtoDisBinary=Ein%NameRoot(1:len_trim(Ein%NameRoot))//"-RrToDis.bim"
705
706     Ut       = UtIn
707     cOutFile = Ein%cSummaryOut
708     write(Us,"(/'ReadYourNml() - done. ',10('|'))")
709                                                                          ;return
710 End Subroutine ReadYourNml
711 !-------------------------------------------------------------------------------7 9
712
713 Subroutine ImportEin(iP)
714 !2025.01.10.1830cst JMS- Generates an Sr record based on:
715 !                       Ein %MtotVis,%mCvis(0:__), & %Vtotshow
716 !--Globals
717  use MS1Def   ,only: Ur,Us,Ut
718  use MS2RCDef ,only: cTitle,RubSize,iType,cFileOut,cLuTOut,EnvNml  & !housekeep
719                     ,Mmax,Mtot,Ctot,C,nS8                          & !Moves
720                     ,Stried8,Sfound8,Vtot,V,Vtotshow,VA,cVAtest    & !Voters
721                     ,H8 ,nRaccum8,Rtot8,SRrec,resetSR,Srseed       & !Sequences
722                     ,Sr1,Srtry,Srbase,Srn,SrTestIn,RVbase          & !& Results
723                     ,nSeqs,nSeqPrev,nSeqPres                       &
724                     ,Ein                                           & !input
725                     ,Csymbol,Asymbol                                 !symbology
726 !--End Globals
727  implicit none
728  !--Arguments
729   integer(4)::iP         !Write enable>5: write(iP,...)
730  !--Internals
731   integer(4)::Vtotl
732   !--EndDefs-----------------------------------------------
733    if(iP>5) write(iP,"(/'Import Ein:     MS3-RC-Workout@L742'/)")
734   !--Internalize the variables in the Ein record:
735    cTitle    = Ein%cTitle
736    cFileOut  = Ein%cSummaryOut
737    cLuTOut   = Ein%cRrAsciiOut
738    iType     = Ein%iType
739    RubSize   = Ein%RubSize
740    Vtot      = Ein%Vtot      !Voters - total
741    V         = 0
742    cVAtest   = Ein%cVAtest
743    V(1:Vtot) = Ein%V(1:Vtot) !Voter(s) | cells          in use
744    Ctot      = Ein%Ctot      !Choices- total
```

```
745     C         = 0
746     C(1:Ctot) = Ein%C(1:Ctot) !Choices   | face rotations in use
747     Mmax      = Ein%Mmax
748     Mtot      = Ein%Mtotvis
749     Vtotshow  = Ein%Vtotshow
750    !--Define Srtest
751     SrTestIn = resetSR  !clears the SrTestIn record
752     SrTestIn%nS8      =       0_8
753     SrTestIn%H8       =       0_8 !undefined
754
755     SrTestIn%Mused    = Ein%Mtotvis
756     SrTestIn%MC(1:30) = Ein%MCvis(1:30)
757     SrTestIn%Vtot     = Ein%Vtot
758              Vtotl    = Ein%Vtot
759     SrTestIn%cVAsymb(1:VtotL)  = Ein%cVAtest(1:VtotL)
760     SrTestIn%VA(     1:VtotL)  = Ein%VA(     1:VtotL)
761
762     call PrintErec(Ein,iP,'ImportEin done.  @L762')
763
764     if(Ein%iType==1) call GenScramble(  Ut)  !in MS6--RC-Solve.f95
765     if(Ein%iType==4) call SolveScramble(Ut)  !in MS6--RC-Solve.f95
766     if(Ein%iType==5) call MinStepSolver(Ut)  !in MS3-Workout.f95
767                      call SaveOutFile
768
769     if(iP>5) write(iP,"('ImportEin done @L769'))")
770                                                                          return
771 End Subroutine ImportEin
772 !-------------------------------------------------------------------------------7 9
773
774 Subroutine PrintErec(Ein,iP,Label)
775 !2025.01.25.1440cst JMS- Compactly reports the content of type(Erec)'s.
776 !--Globals
777  use MS2RCDef, only: Erec,Asymbol,Csymbol !,Rtot8
778 !--End Globals
779  implicit none
780  !--Arguments
781  type(Erec)::Ein              !SRin input record. Label as an "Sr" or "Rr"
782   integer(4)  ::iP              !Write enable>5: write(iP,...)
783   character,optional::Label*(*) !Label length is arbitrary
784  !--Internals
785   integer(4)  ::i,j,nZ
786   !--EndDefs-------------------------------------------------
787    if(iP>5) then
788    write(iP,"(60('-'))")
789    write(iP,"( 'PrintErec: Ein% inputs:    ',\)")
790    if(present(Label))  then
791      write(iP,"('     ""',\)")
792      do i=1,len_trim(Label); write(iP,"(a1,\)") Label(i:i); enddo!i
793      write(iP,"('""')")
794     else; write(iP,*)
795    endif!(present(Label))
796    write(iP,"('    Ein          =',i4,' bytes at runtime')") sizeof(Ein)
797    write(iP,"('   %Einsize      =',i4,'       as written')")Ein%Einsize
798    write(iP,"('   %Zonebimsize  =',i40,            )")Ein%Zonebimsize
799    write(iP,"('   %ZoneFound    =',i40,            )")Ein%ZoneFound
800    write(iP,"('Ein%cTitle        = ',60a1)") &
801                (Ein%cTitle(i:i)  ,i=1,        len_trim(Ein%cTitle)  )
802    write(iP,"('   %NameRoot      = ',60a1)") &
803                (Ein%NameRoot(i:i),i=1,        len_trim(Ein%NameRoot))
804    write(iP,"('   %cSummaryOut   = ',60a1)") &
805                (Ein%cSummaryOut(i:i),i=1,     len_trim(Ein%cSummaryOut))
806
```

```
807     if(Ein%iType/=5) goto 20
808     write(iP,"('   %cRrAsciiOut   = ',60a1)") &
809                 (Ein%cRrAsciiOut(i:i),i=1,     len_trim(Ein%cRrAsciiOut))
810     write(iP,"('   %cSrBinary     = ',60a1)") &
811                 (Ein%cSrBinary(i:i),i=1,       len_trim(Ein%cSrBinary))
812     write(iP,"('   %cRrtoSrBinary = ',60a1)") &
813                 (Ein%cRrtoSrBinary(i:i),i=1,   len_trim(Ein%cRrtoSrBinary))
814     write(iP,"('   %cEinBinary    = ',60a1)") &
815                 (Ein%cEinBinary(i:i),i=1,      len_trim(Ein%cEinBinary))
816     write(iP,"('   %cRrtoDisBinary= ',60a1)") &
817                 (Ein%cRrtoDisBinary(i:i),i=1,  len_trim(Ein%cRrtoDisBinary))
818 20 continue
819
820     write(iP,"('   %iType         =',i4)")                    Ein%iType
821     write(iP,"('   %RubSize       =',i4)")                    Ein%RubSize
822     write(iP,"('   %Vtot          =',i4,' :Cells')")          Ein%Vtot
823
824     write(iP,"('   %V(1:Vtot)     =  ',:,i2,9(',',i2),2(/19x,10(',',i2)))") &
825                                                               Ein%V
826
827     write(iP,"('   %cVAtest       = ""',27a1,'""')") &
828                 (Ein%cVAtest(i:i),i=1,         len_trim(Ein%cVAtest))
829     write(iP,"('   %VA(1:Vtot)    =  ',:,i2,9(',',i2),2(/19x,10(',',i2)))") &
830                                                               Ein%VA
831     do i = 1,20; Ein%cVAtest(i:i) = char(Asymbol(Ein%VA(i))); enddo!i
832     write(iP,"('   %cVAtest       = ""',27a1,'"" -from %VA' )") &
833                 (Ein%cVAtest(i:i),i=1,Ein%Vtot)
834
835     write(iP,"('   %Ctot          =',i4,' :face rotations')")Ein%Ctot
836     write(iP,"('   %C(1:Ctot)     =   ',:,a1,17(',',a1),'   hex')")  &
837                                         Csymbol(Ein%C(1:Ein%Ctot))
838     write(iP,"('   %Mmax          =',i4)")                    Ein%Mmax
839     write(iP,"('   %Mtotvis       =',i4)")                    Ein%Mtotvis
840
841     write(iP,"('   %MCvis(1:',i2,')  =   ',\)") Ein%Mtotvis
842     do i = 1,Ein%Mtotvis
843       if(mod(i,20)/=0) then
844         write(iP,"(a1,      ',',\)") Csymbol(Ein%MCvis(i))
845        else
846        !write(iP,"(a1,i8/,23x,',',\,'***')") Csymbol(Ein%MCvis(i)),i
847         write(iP,"(a1,i8/,20x,',',\      )") Csymbol(Ein%MCvis(i)),i
848       endif!(mod(i,20)/=0)
849     enddo!i
850       write(iP,*)
851     write(iP,"('   %Vtotshow      =',i4)")                    Ein%Vtotshow
852   !-- Zones of Wisdom:
853     write(iP,"(/'   %Ztot           =',i4,' :Wisdom Zones')")Ein%Ztot
854     write(iP,"( '   %Z(1:Ztot)     =  ',:,20i2            )")Ein%Z(1:Ein%Ztot)
855     do i=1,Ein%Ztot; nZ = Ein%Z(i)
856       write(iP,"('   %cZrFilename(',i1,')= ""',:,60a1,'""')") &
857               nZ,(Ein%cZrFilename(nZ)(j:j),j=1        &
858         ,len_trim(Ein%cZrFilename(nZ)) )
859     enddo!nZ
860     write(iP,"('   %nZ            =',i4,' :Zone in use' )")Ein%nZ
861     if(Ein%nZ>0) then
862       write(iP,"('   %ZMaskNeed     =',20i2                )")Ein%ZMaskNeed
863       write(iP,"('   %ZMaskfora     =',20i2                )")Ein%ZMaskfora
864       write(iP,"(' %Vsizez(1:Vtot) = ',20i3                 )")Ein%Vsizez
865       write(iP,"('   %Mtot          =',i4,' :Moves'        )")Ein%Mtot
866     endif!nZ>0
867     endif!(iP>5)
868     write(iP,"(60('-'))")                                                ;return
```

```
869 End Subroutine PrintErec
870 !-------------------------------------------------------------------------7 9
871
872 Subroutine PrintSRrec(SRin,iP,Label)
873 !2024.08.24.1140cdt JMS- Compactly reports the content of type(SRrec)'s.
874 !--Globals
875  use MS2RCDef, only: SRrec,Rtot8,Ein,Csymbol
876 !--End Globals
877  implicit none
878  !--Arguments
879  type(SRrec)::SRin               !SRin input record. Label as an "Sr" or "Rr"
880   integer(4)  ::iP               !Write enable>5: write(iP,...)
881   character,optional::Label*(*) !Label length is arbitrary ~ <= 50 characters
882  !--Internals
883   integer(4)  ::i
884   !--EndDefs-------------------------------------------------
885    if(iP<6)                                                              return
886    write(iP,"(60('-'))")
887    write(iP,"('%nS8    =',i25,'   SRrec: ',\)") SRin%nS8
888    if(present(Label))  then
889      write(iP,"('""',\)")
890      do i=1,len_trim(Label); write(iP,"(a1,\)") Label(i:i); enddo!i
891      write(iP,"('""')")
892     else; write(iP,*)
893    endif!(present(Label))
894    write(iP,"('%H8     =',i25,'   Rtot8  =',i25)") SRin%H8 ,Rtot8
895    write(iP,"('%Mused  = ',i2,2x,'%MC(1:',i2,') =',:,30a2)") &
896                    SRin%Mused,SRin%Mused, char(Csymbol(SRin%Mc(1:SRin%Mused)))
897    write(iP,"('%Vtot   =',i3,'  %cVAsymb  = ',:,20a1)") &
898                       SRin%Vtot, (SRin%cVAsymb(i:i),i=1,SRin%Vtot)
899    write(iP,"('%VA(  ) =',:,20i3)")  SRin%VA(1:SRin%Vtot)
900    if(SRin%nS8==1_8) then
901      write(iP,"('***** %nS8==1 special case: *****')")
902      write(iP,"('SRin%nS8  == 1 has V(1:Vtot) hidden in %MC(2:Vtot+1):')")
903      write(iP, &
904       "('SRin%MC(1       ) =  0, so the record may be used as-is.')")
905      write(iP, &
906       "('SRin%MC(2:Vtot+1) =',:,20i3)") SRin%MC(2:SRin%Vtot+1)
907      write(iP, &
908       "('Ein%V()           =',:,20i3)")  Ein%V( 1:Srin%Vtot  )
909      write(iP, &
910       "('Hence the ""-Sr.bim"" file can stand alone.')")
911      write(iP, &
912       "('SRin%MC(29) =Ctot =',i3)") SRin%MC(29)
913     else
914      write(iP,"('Ein%V() =',:,20i3)")  Ein%V(1:Srin%Vtot)
915    endif!(SRin%nS8==1_8)
916    write(iP,"(60('-'))");                                                 return
917 End Subroutine PrintSRrec
918 !%nS8    =              1   SRrec: "MinStepSolver: Srseed% results @L929:"
919 !%H8     =              1   Rtot8  =       11022480
920 !%Mused  =  0            %MC(1: 0) =
921 !%Vtot   =  7            %cVAsymb  = aaaaaaa
922 !%VA(  ) =  1  1  1  1  1  1  1
923 !*** SRin%nS8  == 1 has V(1:Vtot) hidden in %MC(2:Vtot+1):
924 !    SRin%MC(1       ) =  0, so the record may be used as-is.
925 !    SRin%MC(2:Vtot+1) =  1  2  3  4  5  6  7
926 !    Ein%V()           =  1  2  3  4  5  6  7
927 !    Hence the "-Sr.bim" file can stand alone.
928 !    SRin%MC(29)= Ctot =  9
929 !-------------------------------------------------------------------------7 9
930
```

931