```
1   !MS4-RC-Emulator.f95
2   !2025.01.17.1740cst JMS- Reversion
3   !              Computer- "T4"/HP-800-G4-Mini/i7-8700T/IntelUHD630/Win10Pro-22H2
4   !                         ^name  ^Mfgr.Id     ^chipset ^graphics   ^OS
5   !                            /AbsoftProFortran 21.0.2/OpenGL+Glut3.6
6   !                             ^compiler               ^Fortran graphics
7   !Table of Contents - ToC:
8   !Subroutine EmulatorRC(SRin,nCin,SRtry,valid,iP)   -opens Data-Emulator.txt
9   !Recursive Function Indexer(nin,Srin,iP)           -opens Data-Indexer.txt
10  !---------------------------------------------------------------------------7 9
11
12  Subroutine EmulatorRC(SRin,nCin,SRtry,valid,iP)
13  !2024.08.20.1700cdt JMS- This Rubik's Cube "Emulator" (= model) is exercised by
14  !                        the "Rubik-SER-analysis" program to solve all the
15  !                        3,674,159 valid scrambles of a 2x2x2 Rubik`s Cube
16  !                        in the fewest possible number of moves.
17  !                       (This routine can predict the attitutes of 27 cells.)
18  !--Globals
19   use MS1Def   ,only: Ur,Us,Ut
20   use MS2RCDef ,only: SRrec,AECAV,Rtot8,C,MC                &
21                      ,Ein,RubSize,Sfound8,Vtot,V,Asymbol  &
22                      ,AlV
23  !--End Globals
24   implicit none
25   !--Arguments
26   type(SRrec)::SRin   !A previously sorted (i.e.unique) Rubik`s Cube state.
27    integer(1)::nCin   !the Symbolic move
28   type(SRrec)::SRtry  !The resulting state, sortability not yet determined.
29    integer(4)::valid  !=0:invalid; =nCin:valid
30    integer(4)::iP     !Write enable>5: write(iP,...)
31   !--Internals
32    integer(4)::Init
33   type(SRrec)::SRn    !now copy of SRin, will be modified to become SRtry
34    integer(4)::nV,nV2 !Voter- index
35    integer(1)::nVE    !      = V(nV) is used within the model- cell#
36    integer(4)::nA,nA2 !Choice- index
37   !integer(4)::nA3,nA4
38  !!integer(4)::nC !Choice- index
39    integer(1)::nCE    !      = C(nC) is used within the model- rotation#
40    integer(1)::Mused  !Moves - total used thusfar
41    integer(1)::VAn(27)!now copy of SRn%VA()
42   !integer(4)::i,nAl
43    !EndDefs---------------------------------------
44     if(Init==0) then
45       write(Us,"('EmulatorRC:Import Data-Emulator.txt:AECAV18(0:18,24,27):')")
46
47       if(iP>5) write(iP,"(/,'EmulatorRC(SRin...) initializing @L50:')")
48      !-- Import AECAV:
49       if(iP>5)write(iP,"('Import AECAV(0:18,24,27): 2024.08.05.1729:')")
50       open(11, file='Data-Emulator.txt', action='read' &
51              , access='sequential'      , status='old'    )
52         read(11,"(////)")  !Skips the first 5 lines
53        !write(13,"(/'            | D  |  E  |  F  |  G  |  H  |  I  |' )")
54  !!     write(13,"( '          0  1  2  3  4  5  6  7  8  9  A  B  C',\)")
55  !!     write(13,"('| D  E  F  G  H  I  _'/)")
56         do nV = 1,27
57           do nA = 1,24
58             read(11,*) nV2,nA2,AECAV(0:19,nA2,nV2)
59            !write(13,"(2i4,':',20i3)") nV2,nA2,AECAV(0:19,nA2,nV2)
60           enddo!nA
61         enddo!nV
62       close(11)
```

```fortran
63       Init = 1
64     endif!(Init==0)
65
66    !--Normal entry point after initialization:
67     if(iP>5) &
68     write(iP,"(/,'EmulatorRC: nCin =',i4,'  %Va = ',:, 20a1)") &
69                                nCin,      char(Asymbol(SRin%VA(1:Vtot)))
70     valid = 0
71     if(RubSize==2) then !Check Choice index validity & define nCE:
72       if((nCin<1).or.(nCin> 9)) goto 90  !Report error
73      else
74       if((nCin<1).or.(nCin>18)) goto 90  !Report error
75     endif!(RubSize==2)
76    !nCin is in range, proceed:
77     nCE = C(nCin)
78
79    !The model interfaces with record SRn()%*:
80     SRn           = SRin  !SRn becomes SRtry
81         Mused     = SRin%Mused+1
82         nCE       = C(nCin)
83         VAn(1:20) = SRn%VA                                    !2024.08.17
84
85     SRn%nS8       = Sfound8+1_8
86
87     SRn%Mused     = Mused
88  !!!SRn%MC(  0  ) = Mused
89     SRn%MC(Mused) = nCE
90
91    !////////////////////////////////////////////////
92    !Modelling- cycle through all the active cells:
93     do nV = 1,Vtot;   nVE = V(nV)
94      !Here's the beef: the face rotation changes one cell's attitude:
95       SRn%VA(nV) = AECAV(nCE,VAn(nV),nVE)
96      !       ... & the cell`s attitude character symbol is updated:
97       SRn%cVAsymb(nV:nV) = char(Asymbol(SRn%VA(nV)))
98      !Modelling completed. ----------------------- in four lines of Fortran
99     enddo!nV
100   !////////////////////////////////////////////////
101
102    SRtry = SRn; valid = nCE
103    if(iP>5)then
104     !call PrintSRrec(SRin ,iP,'SRin' )
105      write(iP,"(6x,'nCE = ',i2)") nCE
106     !call PrintSRrec(SRtry,iP,'SRtry')
107    endif!(iP>5)
108                                                                          return
109 90 continue
110    call SaveOutFile
111    pause 'EmulatorRC error @L113'
112                                                                          return
113 End Subroutine EmulatorRC
114 !---------------------------------------------------------------------------7 9
115
116 Recursive Function Indexer(nin,Srin,iP) Result(Out)  !<-integer(8)
117 !2024.08.20.1720cdt JMS- Rubik`s Cube Results address id function.
118
119 !--Globals
120  use MS1Def   ,only: Ur,Us,Ut
121  use MS2RCDef ,only: SRrec,V,Ein,AlV,resetAlV,RVbase,RV  & !,Vtot
122                     ,RubSize,H8 ,r16H8 ,nRaccum8,r16accum8,RaidInit,Ctot
123 !--End Globals
124  implicit none
```

```
!--Arguments
 integer(4)::nin    !Recursion Level index- `called by` level
type(SRrec)::Srin    !id nr8 of this Sr record
 integer(4)::iP      !Write enable>5: write(iP,...)
 integer(8)::Out     !function result
!--Internals
 integer(4)::n       !Recursion Level index- of this level
                     !  ...passed downward but not upward
 integer(4)::nV      !Voter/cell in use
 integer(4)::i,nA,nA2,nAl,nV2
 integer(4)::nAavail,nAcount
!integer(4)::nNew
 integer(8)::H8a
 real(16)  ::r16Check
 character(22)::Datime22L
 integer(4)::Vtot

 !--EndDefs---------------------------------------------------
    if(RaidInit==0) then

    if(iP>5) write(iP,"('Indexer: Import Data-Indexer.txt:  Alp(1:20):')")

   !-- Import AlV(1:20):
    if(iP>99)write(iP,"('Import AlV(nV): 2024.07.17.2055:')")
    open(Ur, file='Data-Indexer.txt', action='read' &
           , access='sequential'     , status='old'    )
      read(Ur,"(////)") !Skips the first 5 lines
      do nV = 1,8
        read(Ur,*) nV2,AlV(nV2)%Al(1:24); AlV(nV2)%nV = nV2
                       AlV(nV2)%nAl(1:24) = ichar(AlV(nV2)%Al(1:24))-96
        do i=1,20;     AlV(nV2)%lavail(i) = i
        enddo!i
       do i=1,24;                  nAl    = AlV(nV2)%nAl(i)
                       AlV(nV2)%nAu(nAl)  = nAl  ;enddo!i
      enddo!nV
      read(Ur,"(/)")    !Skips 2 line
      do nV = 9,20
        read(Ur,*) nV2,AlV(nV2)%Al(1:24); AlV(nV2)%nV = nV2
                       AlV(nV2)%nAl(1:24) = ichar(AlV(nV2)%Al(1:24))-96
        do i=1,20;     AlV(nV2)%lavail(i) = i    ;enddo!i
        do i=1,24;                  nAl    = AlV(nV2)%nAl(i)
                       AlV(nV2)%nAu(nAl)  = nAl  ;enddo!i
      enddo!nV
      read(Ur,"(///)")  !Skips the 4 lines
      do nV = 1,8
        read(Ur,*) nV2,AlV(nV2)%mdis(1:24)
      enddo!nV
      read(Ur,"(/)")    !Skips 2 line
      do nV = 9,20
        read(Ur,*) nV2,AlV(nV2)%mdis(1:24)
      enddo!nV
      read(Ur,"(////)")  !Skips the 5 lines
      do nV = 1,20
        read(Ur,*) nV2,AlV(nV2)%nAloc(1:24)
      enddo!nV
    close(Ur)

    if(iP>5) then
      write(iP,"(/' n, RVbase(n)%nAloc(1:24):')")
      write(iP,"( 5x,'a  b  c  d  e  f  g  h  i  j  k  l  m  n  o  p  q  r',\)")
      write(iP,"(  '  s  t  u  v  w  x')")
    endif!(iP>5)
```

```
        RVbase              = resetAlV
      do n = 1,Ein%Vtot;           nV = Ein%V(n)
        RVbase(n)              = AlV(nV)
        RVbase(n)%n            = n            !RV#
        RVbase(n)%nV           = nV           !Cell#
        if(iP>5) write(iP,"(i2,': ',i2, 23(',',i2) )")  &
                    n,    (RVbase(n)%nAloc(i),i=1,24)
      enddo!n
      RVbase(1:Ein%Vtot)%nA = Ein%VA(1:Ein%Vtot)  !cell Attitude#'s
      if(iP>5) write(iP,"('Indexer: Global initialization completed',/)")
      RaidInit = 1
    endif!(RaidInit==0) -------------------------

    n = nin+1
    if(n==1) then  !n==1 is the entry level of the recursion process.
   !--Reinitialize the recursive addressing process:
     !write(Us,"(/'Indexer: Recursive initialization:')")
      if(iP>5) write(iP,"(//'Find the next address recursively:',/)")
        Vtot                   = Ein%Vtot
        RVbase                 = resetAlV
      do n = 1,Ein%Vtot;           nV = Ein%V(n)
        RVbase(n)              = AlV(nV)
        RVbase(n)%n            = n            !RV#
        RVbase(n)%nV           = nV           !Cell#
        if(iP>5) write(iP,"(i2,': ',i2, 23(',',i2) )")  &
                    n,    (RVbase(n)%nAloc(i),i=1,24)
      enddo!n
      RVbase(1:Vtot)%nA = Ein%VA(1:Vtot)  !cell Attitude#'s
      n = 1

      H8              = 0_8
      nRaccum8        = 0_8
      RV              = RVbase
      RV(1:Vtot)%nA = Srin%VA(1:Vtot) !Reinitializing the attitudes
      RV(0)     = RV(1)
                              nV = Ein%V(n)
      RV(0)%nV              = nV
      if(RubSize==2) &
      RV(0)%lavail(8:20) = 0

      if((RubSize==3).and.((Ctot==6).or.(Ctot==9))) then  !2024.12.23 - no 180
         RV(0)%lavail( 8) = 0
         RV(0)%lavail(16) = 0
         RV(0)%lavail(19) = 0
         RV(0)%lavail(20) = 0
      endif!((RubSize==3).and.(Ctot==9))

      if(iP>5) then
      write(iP,"(/'Indexer   :  n nV')")
      write(iP,"( '...start :  n = ',i3,'   Indexer() recursive initialization')") n
      write(iP,"(/'RubSize       = ',i2)") RubSize
      write(iP,"( 'Vtot          = ',i2)") Vtot
      write(iP,"( 'RV(1:Vtot)%nA = ',:,i2, 23(',',i2))") RV(1:Vtot)%nA
      write(iP,"( 'RP( 0)%lavail = ',:,i2, 19(',',i2))") RV(0)%lavail
      endif!(iP>5)
     !write(Vs,"(/'Indexer: Recursive initialization completed')")
    endif!(n==1)

    RV(n)%lavail = RV(n-1)%lavail
    if((n>1).and.( RV(n-1)%nLused>0)) &
      RV(n)%lavail(RV(n-1)%nLused) = 0
                                                   nV = RV(n)%nV
```

```fortran
                                                     nA = RV(n)%nA
   if(iP>5) then
     write(iP,"(/'n  = ',i2,' -------------------')") n
     write(iP,"( 'nV = ',i2                        )") nV
     write(iP,"( 'nA = ',i2                        )") nA
     write(iP,"('RV(',i2,')%nlused = ',i2)")               n-1,RV(n-1)%nlused
     write(iP,"('RV(',i2,')%lavail = ',:,i2, 19(',',i2))") n  ,RV(n  )%lavail

     select case(nV)
      case(1: 8)
       write(iP,"('1:8  locations / still open / RV(',i2,')%nAl:'))") n
       do i = 1, 8; write(iP,"(i5,'   |',\)")              i ;enddo!i
                    write(iP,*)
       do i = 1, 8; write(iP,"(i5,'   |',\)") RV(n)%lavail(i);enddo!i
                    write(iP,*)
!1: 8 locations:|    1   |    2   |    3   |    4   |    5   |    6   |    7   |
!    still open:|    1   |    2   |    3   |    4   |    0   |    6   |    7   |
!RV( 2)%nAl    =  9,15,17, 3, 5,23,21,22,24, 8,16,18, 4, 6,19, 1,11,14,10,12,13, 2,
      case(9:20)
       write(iP,"('9:20 locations: / still open / RV(',i2,')%nAl:'))") n
       do i = 9,20; write(iP,"(i3,'  |' ,\)")              i ;enddo!i
                    write(iP,*)
       do i = 9,20; write(iP,"(i3,'  |' ,\)") RV(n)%lavail(i);enddo!i
                    write(iP,*)
!9:20 locations:|  9  | 10  | 11  | 12  | 13  | 14  | 15  | 16  | 17  | 18  | 19  |
!    still open:|  9  | 10  | 11  | 12  | 13  | 14  | 15  | 16  |  0  | 18  | 19  |
!RV( 4)%nAl    = 14,24, 4, 8, 9,20,13,23, 3,12,18,19, 2,17,11,22, 7,15, 1,21, 5,10,
     end select!(nV)

     write(iP,"(                       :,i2, 23(',',i2))")     RV(n  )%nAl
   endif!(iP>5)

   RV(n)%nLused = RV(n)%nAloc(nA)
   if(iP>5) &
   write(iP,"('RV(',i2,')%nLused = ',i2)")                 n  ,RV(n  )%nLused

  !List the Attitudes available:
   RV(n)%nAu     = 0
         nAcount = 0;    nAavail   = 0
   select case(nV) !----------
    case(1: 8) !corner cells:
     if((RubSize==2).and.(nV==8)) goto 20
     do nA2 = 1,24
       if(RV(n)%lavail(RV(n)%nAloc(nA2))==0) cycle
                   nAavail   =  nAavail+1
       RV(n)%nAu(  nAavail) = nA2
       if(nA2==nA) nAcount   = nAavail
     enddo!nA2
20   continue
    case(9:20) !edge cells:
     do nA2 = 1,24
       if(RV(n)%lavail(RV(n)%nAloc(nA2))==0) cycle
                   nAavail   =  nAavail+1
       RV(n)%nAu(  nAavail) = nA2
       if(nA2==nA) nAcount   = nAavail
     enddo!nA2
   end select!nP !----------
   RV(n)%nAavail = nAavail
   RV(n)%nAcount = nAcount

   if(iP>5) then
   write(iP,"('RV(',i2,')%nAu     = ',:,i2, 23(',',i2))") n  ,RV(n  )%nAu
```

```fortran
311    write(iP,"('RV(',i2,')%nAavail = ',i2)")               n,RV(n)%nAavail
312    write(iP,"('RV(',i2,')%nAcount = ',i2)")               n,RV(n)%nAcount
313    write(iP,"( 'pre-recur:',3i3,i22)") n,nV,RV(n)%nAavail,RV(n)%nAcount
314    endif!(iP>5)
315
316   !--the recursion:
317    if(n.lt.Ein%Vtot) H8a = Indexer(n,Srin,iP)       ! <<<<<----- *****
318
319   !--at--full-recursive-depth calculations:
320    if(n==Ein%Vtot)then
321            H8  = 1_8   !this offsets the subsequent -1 subtraction @L325
322         r16H8  = 1._16 !counts to ~ 5.0e30
323       nRaccum8 = 1_8   !counts to ~ 9.2e18
324      r16accum8 = 1._16 !counts to ~ 5.0e30
325      if(iP>5) &
326      write(iP,"( 'full-in  :',2i3,3x,i22,12x,i22 )") n,nV,nRaccum8,H8
327
328    endif!(nV==4)
329
330   !--de-recursion:
331       H8     =    H8  + (RV(n)%nAcount-1) *  nRaccum8
332     nRaccum8 =           RV(n)%nAavail    *  nRaccum8
333
334    r16H8     = r16H8  + (RV(n)%nAcount-1) * r16accum8
335    r16accum8 =           RV(n)%nAavail    * r16accum8
336
337    if(iP>5) then
338      r16Check = r16accum8-nRaccum8
339      if(abs(r16Check)<.5_16) then     !nRaccum8 in error by less than .5
340        write(iP,"( 'de-recur :',3i3,i22,'   *',i3,' + ^> =',i22)") &
341          n,nV,RV(n)%nAavail,nRaccum8,(RV(n)%nAcount-1),H8
342       else
343        write(iP,"( 'de-recur :',3i3,f24.1,' *',i3,' + ^> =',f24.1)") &
344          n,nV,RV(n)%nAavail,r16accum8,(RV(n)%nAcount-1),r16H8
345      endif!(abs(r16Check)<.5_16)
346
347    endif!(iP>5)
348
349    if(n==1) then
350      if(iP>5)write(iP,"(8x,2('          \sx\qn\qd\tr\bl\ml\th\hn\'))")
351      if(iP>5)write(iP,"(9x,2('         ^25^22^18^15^12^9 ^6 ^3 ^0'))")
352      if(abs(r16Check)>.5_16) then
353        if(iP>6) then
354          call jdate22(DaTime22L)
355          write(iP, &
356       "(/'***** Integer(8) overflowed: @L361 *****',17x,a23,/)") DaTime22L
357          call SaveOutFile
358        endif!(iP>6)
359        pause '***** Integer(8) overflowed. @L364. Press enter to continue.'
360       !stop
361      endif!(abs(r16Check)<.5_16)
362  !--Report the address on recursion exit
363      Srin%H8   = H8
364      Out       = H8
365    endif!(n==1)
366                                                                              return
367  End Function Indexer
368 !----------------------------------------------------------------------------7 9
369
```