

# riset Group

John D. Baker

<https://github.com/bakerjd99/jackshacks/blob/main/risetset.ijs>

SHA-256: b3e4d29044b7d331e713e2d587c8d38dad69641990bb74df267db4181fe55a40

May 19, 2023

## Contents

<b>riset Overview</b>	<b>2</b>
riset Interface . . . . .	3
riset Algorithm Notes . . . . .	3
<b>riset Source Code</b>	<b>6</b>
<b>=: Index</b>	<b>56</b>

## risetset Overview

`risetset` is a collection of basic astronomical algorithms that compute the rise, transit, and set times of [IAU-named](#) stars.

`risetset` is distributed as an auxillary J addon. Auxillary addons are hosted in private GitHub repositories. `risetset` can be installed in the local J folder `~addons/jacks` with:

```
load 'pacman'
```

*NB. files from <https://github.com/bakerjd99/jackshacks>*

```
install 'github:bakerjd99/jackshacks'
```

*NB. installed files*

```
dir '~addons/jacks'
```

Once installed it can be loaded and run with:

```
load '~addons/jacks/risetset.ijs'
```

```
location_yellowstone 0
```

*NB. IAU stars rising/setting over Old Faithful*

```
fmt_today iau_today 0
```

For more details about setting up, running, and “hacking” `risetset` refer to the Jupyter notebook [risetset\\_notebook.ipynb](#) or the pdf version of same [risetset\\_notebook.pdf](#). Both of these files are installed with `risetset`.

## riseset Interface

`baby_today` [13] *named Babylonian stars rising/setting today*  
`fmt_today` [21] *format today verbs result*  
`iau_today` [23] *named IAU stars rising/setting today*  
`loadstars` [26] *loads riseset star data*  
`nav_today` [35] *named navigation stars rising/setting today*  
`navdaylist` [36] *sky safari 6\_0 observing list of today's navigation stars*  
`riseset` [44] *rise, transit, set times of stars*

## riseset Algorithm Notes

Many `riseset` algorithms are taken from Jean Meeus's book *Astronomical Algorithms*. A PDF copy of this book is available [here](https://ia802807.us.archive.org/20/items/astronomicalalgorithmsjeanmeeus1991/):

<https://ia802807.us.archive.org/20/items/astronomicalalgorithmsjeanmeeus1991/>

also here:

[https://www.amazon.com/s?i=stripbooks&rh=p\\_27%3AJean+Meeus&s=relevancerank&text=Jean+Meeus&ref=dp\\_byline\\_sr\\_book\\_1](https://www.amazon.com/s?i=stripbooks&rh=p_27%3AJean+Meeus&s=relevancerank&text=Jean+Meeus&ref=dp_byline_sr_book_1)

Nutation algorithms are from Jay Tanner's site:

<https://neoprogrammics.com/nutations/index.php>

Markdown versions of Tanner's algorithms are stored in the `JOD futs` and `utils` dictionaries — see:

1. `nutaton_in_longitude_dPsi_md`
2. `nutaton_in_obliquity_of_ecliptic_dEpsDeg_md`

in:

<https://github.com/bakerjd99/joddumps/blob/master/futs.ijs>

<https://github.com/bakerjd99/joddumps/blob/master/utils.ijs>

Delta T ( $\Delta T$ ) is computed using polynomial expressions by Espenak and Meeus, see:

<https://eclipse.gsfc.nasa.gov/SEhelp/deltatpoly2004.html>

A markdown version of the Delta T ( $\Delta T$ ) algorithm is in `futs` see:

#### 1. nasa\_polynomial\_expressions\_for\_delta\_t\_md

The sunrise and set verb `sunriseset1` is a J version of a BASIC program featured in the March 3 1995 *Astronomical Computing* column of *Sky and Telescope*. The Free Library maintains a copy of this article - see:

<https://www.thefreelibrary.com/Sunrise+sunset+challenge%3a+the+winners.-a016565215>

<https://skyandtelescope.org/about-us/sky-and-telescope-index/>

Many other interesting BASIC *Sky and Telescope* programs are here:

<https://skyandtelescope.org/astronomy-resources/basic-programs-from-sky-telescope/>

You can display the markdown referenced about with the JOD expressions:

```
load 'general/jod'
od ;:'futs utils'
NB. display markdown documents
4 disp ; }.@(4&dn1)&.> 'nutation_';'nasa_'
```

Many examples showing how to use various `riseset` words are in the JOD `futs` test suite `riseset`. You can display all the test cases with the JOD expressions:

3 grp 'riseseet'    *NB. test cases in suite*

3 disp 'riseseet'    *NB. display test suite*

4 rtt 'riseseet'    *NB. run all tests in suite*

## riset Source Code

```
NB.*riset s-- compute rise, transit and set times of IAU named stars.
NB.
NB. verbatim: interface word(s):
NB. -----
NB.  baby_today - named Babylonian stars rising/setting today
NB.  fmt_today  - format today verbs result
NB.  iau_today   - named IAU stars rising/setting today
NB.  loadstars  - loads riset star data
NB.  nav_today   - named navigation stars rising/setting today
NB.  navdaylist - sky safari 6_0 observing list of today's navigation stars
NB.  riset      - rise, transit, set times of stars
NB.
NB. created: 2023mar09
NB. changes: -----
NB. 23mar29 (iau_tonight) renamed (iau_today)
NB. 23mar29 various location setting verbs (location_uluru) added
NB. 23mar30 (nav_today) added
NB. 23apr01 (fmt_today) added
NB. 23apr06 (navdaylist) added
NB. 23apr08 (baby_today) added
NB. 23apr27 show sunrise/set times added (localsun)

coclass 'riset'

(9!:11) 16 NB. high print precision
```

*NB.\*end-header*

*NB. carriage return character*

CR=: 13{a.

*NB. minutes before and after sunset (0=ignore sun)*

DARKTRS=: 60

*NB. seconds per day*

DAYSECS=: 86400

*NB. interface words (IFACEWORDSriseset) group*

IFACEWORDSriseset=: <;.\_1 ' baby\_today fmt\_today iau\_today loadstars nav\_today navdaylist riseset'

*NB. current Julian date*

JULIAN=: 2460030.5

*NB. line feed character*

LF=: 10{a.

*NB. horizon limit in degrees*

LIMITHORZ=: 20

*NB. limiting magnitude*

LIMITMAG=: 3.

*NB. Name/description of observer location*

LOCATIONNAME=: 'Meridian'

*NB. indicates sun never rises or sets in (sunriseset0) and (sunriseset1) results*

NORISESET=: 99

*NB. approximate epoch J2000 obliquity of the ecliptic degrees, minutes, seconds*

OBLIQUITYDMS2000=: 23 26 21.4480000000000004

*NB. observer latitude longitude, west longitudes negative*

OBSLOCATION=: \_116.375956000000002 43.6467749999999981

*NB. root words (ROOTWORDSriseset) group*

ROOTWORDSriseset=: <;.\_1 ' IFACEWORDSriseset ROOTWORDSriseset VMDriseset baby\_today fmt\_today iau\_today loc  
>..>ation\_uluru location\_yellowstone navdaylist'

*NB. standard altitude stars - compensates for horizon atmospheric refraction*

STDALTITUDE=: 0.566699999999999982

*NB. UTC time zone offset in hours*

UTCOFFSET=: 6

*NB. version, make count and date*

VMDriseset=: '0.9.85';2;'19 May 2023 10:44:16'



*NB. retains string after first occurrence of (x)*

```
afterstr=: ] }.~ #@[ + 1&(i.~)@([ E. ])
```

*NB. all zero, first, second, ... nth differences of nl: alldifs ?.10#100*

```
alldifs=: ([: >: [: i. [: - #) {.&.> [: <"1 (}. - }:)^(i.@#@[)
```

```
apparRADEC=: 4 : 0
```

*NB.\*apparRADEC v-- apparent RA and DEC for epoch (x) from J2000.0*

*NB. RA and DEC.*

*NB.*

*NB. This verb adjusts J2000 RA and DEC coordinates to another*

*NB. epoch. The method is based on Meeus (20.3) pg 126. This*

*NB. calculation ignores stellar proper motions and assumes that*

*NB. (y) RA DEC values are J2000.0. The resulting positions are*

*NB. accurate enough for basic rise, transit, and set*

*NB. calculations.*

*NB.*

```
NB. dyad: ft =. flymd apparRADEC ftRADEC
```

*NB.*

```
NB. 2028 11 13.19 apparRADEC 41.054063 ,. 49.227750
```

*NB.*

```
NB. ({."1 ciau)=: {:"1 ciau
```

```
NB. 2023 4 22 apparRADEC RA_J2000 ,: Dec_J2000
```

```
'zet z th'=. zetzthT0 x NB. final epoch t
```

```
'ra dec'=. y NB. J2000 ra,dec
```

*NB. meeus (20.4) pg. 126*

```
A=. (cosd dec)*sind ra + zet
B=. ((cosd th)*(cosd dec)*cosd ra + zet) - (sind th)*sind dec
C=. ((sind th)*(cosd dec)*cosd ra + zet) + (cosd th)*sind dec
```

*NB. NIMP star close celestial poles*

*NB. new dec,ra*

```
ran=. z + atan2 A ,: B [ decn=. dfr arcsin C
ran ,: decn
)
```

*NB. seconds correction apparent sidereal time - meeus pg. 84 - ( $\Delta\psi \cdot \cos(\epsilon)$ )/15*

```
apparsecs=: 15 %~ (3600 * nutation_longitude_dPsi) * [: cosd meanobliquityjd0
```

*NB. apparent Greenwich sidereal - hms: apparsidjd0 julfrcal |: 2023 1 3,:1991 2 8.5*

```
apparsidjd0=: ([: dmsfrdd 15 %~ [: nth0 meansidjd0) + 0 0 ,"1 [: ,. apparsecs
```

*NB. applies the verb in string (x) to (y)*

```
apply=: 128!:2
```

*NB. arc cosine*

```
arccos=: _2&o.
```

*NB. arc sine*

```
arcsin=: _1&o.
```

*NB. arc tangent*

arctan=: \_3&o.

*NB. signal with optional message*

assert=: 0 0"\_ \$ 13!:8^((0: e. ])^(12"\_))

atan2=: 3 : 0

*NB.\*atan2 v-- arctangent of (Y % X) in degrees.*

*NB.*

*NB. FORTRAN (ATN2) variation of the standard (arctan) (\_3&o.) for*

*NB. ratios. Based on a PASCAL function from Astronomy on the*

*NB. Personal Computer by Montenbruck and Pfleger ISBN*

*NB. 0-387-52754-0 pg. 9.*

*NB.*

*NB. Result is between \_180 <: atan2 <: 180 degrees*

*NB.*

*NB. monad: fl =. atan2 flYX*

*NB.*

*NB. atan2 1 ,: 1 NB. 45 degrees*

*NB. atan2 1 ,: %: 3 NB. 30 degrees*

*NB.*

*NB. NB. random ratios comparing two atan2 verbs*

*NB. r=: ?. 2 500\$50*

*NB. r=: r \* (\$r) \$ (?.~ \*/\$r) { (\*/\$r)\$\_1 1*

*NB. (atan2b |.r) -: atan2 r*

*NB.*

```

NB. NB. surprisingly (atan2) is faster than (atan2b)
NB. NB. (j 9.41 2023) but (atan2b) consumes less memory
NB. NB. 1000 ts"1 'atan2b r',: 'atan2 |.r'

NB. vector J                      NB. scalar PASCAL
rad=. 0.0174532925199432955  NB. CONST RAD=0.0174532925199433;
r=. 0 #~ {: $y

b0=. */0=y                      NB. IF (X=0.0) AND (Y=0.0) THEN ATN2:= 0.0
ir=. i. #r=. 0 (I. b0)} r

if. +./b1=. -.b0 do.
  t=. |(I. b1) {"1 y          NB. AX=: ABS(X); AY=: ABS(Y)
  it=. (I. b1) { ir

  b2=. (1{t) > 0{t            NB. IF (AX>AY) THEN PHI=: ARCTAN(AY/AX)/RAD
  s=. (I. b2) {"1 t
  r=. (rad %~ arctan %/s) (b2#it)} r

  s=. (I. -.b2) {"1 t          NB. ELSE PHI=: 90.0-ARCTAN(AX/AY)/RAD;
  r=. (90 - rad %~ arctan %/ |.s) (it #~ -.b2)} r
end.

x10=. I. b1 *. (1{y) < 0       NB. IF (X<0.0) THEN PHI=: 180.0-PHI;
r=. (180 - x10{r) (x10)} r
y10=. I. b1 *. (0{y) < 0       NB. IF (Y<0.0) THEN PHI=: -PHI;
(-y10{r) (y10)} r

```

```
)

baby_today=: 3 : 0

NB.*baby_today v-- named Babylonian stars rising/setting today.
NB.
NB. monad: (bt ; clLoc ; itRs ; flParms) =. baby_today uuIgnore

jd=. julfrcal ymd=. 3 {. 6!:0 ''
(ymd;jd;OBSLOCATION;UTCOFFSET;LIMITMAG;LIMITHORZ;LOCATIONNAME;DARKTRS) baby_today y
:
NB. star data
({."1 IAU)=. {:"1 IAU [ 'IAU NAV'= . loadstars 0
bs=. babylonian_named_stars 0

NB. !(*)=. IAU_Name Designation
'Rs lName sRs cParms'= . x today_calc }. 0 {"1 bs
NB. include Designation names
Rs=. 1 0 2 3 {"1 Rs ,.~ (IAU_Name i. 0 {"1 Rs){Designation
Rs;lName;sRs;cParms
)

babylonian_named_stars=: 3 : 0

NB.*babylonian_named_stars v-- identified Babylonian stars approx
NB. 1500 BCE.
NB.
```

```
NB. Stars with modern names identified from ancient Babylonian
NB. tablets. Most stars will be shining long after we are gone.
NB. It's fun to seek out stars that the ancients found important
NB. enough to catalog. Source data comes from a spreadsheet TAB
NB. here:
NB.
NB. https://www.iau.org/public/themes/naming\_stars/
NB.
NB. monad: bt=. babylonian_named_stars uuIgnore

NB. load babylonian stars !(*)=. HIP IAU_Name jpath
bs=. parsebomcsv read jpath '~addons/jacks/testdata/babylonian_normal_stars.csv'

NB. cross reference with current names
(0 {"1 ciau)=. 1 {"1 ciau [ 'ciau cnavs'= . loadstars 0
bs=. bs #~ 1,HIP e.~ }. 0 {"1 bs
ix=. HIP i. }. 0{"1 bs
bs=. ('IAU_Name';ix{IAU_Name) ,. bs

NB. remove columns without names
bs #"1~ ] 0 < #&> 0 { bs
)

NB. retains string before first occurrence of (x)
beforestr=: ] {~ 1&(i.~)@([ E. ])

NB. boxes open nouns
boxopen=: <^(L. = 0:)
```

```
cold_iau_named_stars=: 3 : 0
```

```
NB.*cold_iau_named_stars v-- convert IAU btcl to column dictionary.
```

```
NB.
```

```
NB. monad: bt =. cold_iau_named_stars btcl
```

```
NB.
```

```
NB. iau=. ; {: , > {: 4 get 'iau_named_stars_2022_txt'
```

```
NB. ciau=. cold_iau_named_stars parse_iau_named_stars iau
```

```
NB.
```

```
NB. NB. define columns
```

```
NB. (0 {"1 ciau)=: 1 {"1 ciau
```

```
c=. 0{"1 t=. |: y
```

```
p0=. c i. ;:'Vmag RA_J2000 Dec_J2000'
```

```
d=. _999&".&> p0 { t=. }. "1 t
```

```
'invalid mag, ra, dec' assert -. _999 e. d
```

```
p1=. c i. ;:'IAU_Name Designation HIP Bayer_Name'
```

```
c ,. (<"1 ] p1 { t) , <"1 d
```

```
)
```

```
NB. cosine radians
```

```
cos=: 2&o.
```

```
NB. cosine degrees
```

```
cosd=: cos@rfd
```

```
NB. character table to newline delimited list
```

```
ctl=: }.@(.@1&(",1)@(-.@(*./\."1@(&' '[@])))) # ,@((10{a.)&(",1)@[]))
```

```
darktransits=: 4 : 0
```

```
NB.*darktransits v-- mask selecting transits before and after sunset.
```

```
NB.
```

```
NB. dyad: pl =. itHrmn darktransits (itSrs ; iaMins)
```

```
NB.
```

```
NB. 'Riseset Location cParms'=. (location_yellowstone~ 1935 7 6) nav_today 0
```

```
NB. srs=. localsun 1935 7 6
```

```
NB. (>{: "1 Riseset) darktransits srs;60
```

```
NB. sun rise/set in day minutes - dark minutes
```

```
'srs bam'=. y
```

```
if. (NORISESET,1) -: 0{srs do. 0 #~ #x NB. sun is always up
```

```
elseif. (NORISESET,0) -: 0{srs do. 1 #~ #x NB. sun is always down
```

```
elseif. do.
```

```
NB. transit times in day minutes and before/after set minutes
```

```
rs=. dmfrhm x [ 'sr ss'=. dmfrhm srs
```

```
NB. transits occurring when sufficiently dark
```

```
(rs < 0 >. sr - bam) +. rs > 1440 <. ss + bam
```

```
end.
```

```
)
```

```
NB. decimal degrees from degrees, minutes, seconds - inverse (dmsfrdd)
```

```
ddfrdms=: (60"_ #. ]) % 3600"_
```



```
deltaT0=: 3 : 0
```

```
NB.*deltaT0 v-- dynamical time  $\Delta T$  in seconds.
```

```
NB.
```

```
NB. Returns the difference in seconds between UT and TD based on
```

```
NB. polynomial expressions by Espenak and Meesus. This
```

```
NB. calculation is useful for the years -1999 to 3000: a five
```

```
NB. thousand year period.
```

```
NB.
```

```
NB. see: https://eclipse.gsfc.nasa.gov/SEhelp/deltatpoly2004.html
```

```
NB.
```

```
NB. also in (futs): nasa_polynomial_expressions_for_delta_t_md
```

```
NB.
```

```
NB. monad: flSecs =. deltaT0 flYd
```

```
NB.
```

```
NB. ymd=. |: (3 {. 6!:0 ' '), _1812 3 12 , _12 12 11 , 2137 12 13, 1700 1 1 ,: 35 7 6
```

```
NB. |: ymd , deltaT0 deltaTdy ymd
```

```
NB. (ry) time intervals are (l,u]
```

```
NB. before -500:
```

```
NB.  $\Delta T = -20 + 32 * u^2$ ; where:  $u = (y-1820)/100$ 
```

```
ry=. ,: _1999 _500
```

```
t1=. {{ _20 + 32 * U^2 [ U=. (y - 1820) % 100 }}
```

```
NB. between -500 and +500:
```

```
NB.  $\Delta T = 10583.6 - 1014.41 * u + 33.78311 * u^2 - 5.952053 * u^3$ 
```

```

NB.      - 0.1798452 * u^4 + 0.022174192 * u^5 + 0.0090316521 * u^6; where: u = y/100
NB.  NOTE: for the year -500 set value of 17190 to 17203.7
ry=. ry , _500 500
t2=. {{ 10583.6 - (1014.41*U) + (33.78311*U^2) - (5.952053*U^3) - (0.1798452*U^4) + (0.022174192*U^5) + 0.
>..>0090316521*U^6 [ U=. y % 100 }}

NB. between +500 and +1600:
NB.   $\Delta T = 1574.2 - 556.01 * u + 71.23472 * u^2 + 0.319781 * u^3$ 
NB.      - 0.8503463 * u^4 - 0.005050998 * u^5 + 0.0083572073 * u^6; where: u = (y-1000)/100
ry=. ry , 500 1600
t3=. {{ 1574.2 - (556.01*U) + (71.23472*U^2) + (0.319781*U^3) - (0.8503463*U^4) - (0.005050998*U^5) + 0.00
>..>83572073*U^6 [ U=. (y-1000) % 100 }}

NB. between +1600 and +1700:
NB.   $\Delta T = 120 - 0.9808 * t - 0.01532 * t^2 + t^3 / 7129$ ; where: t = y - 1600
ry=. ry , 1600 1700
t4=. {{ 120 - (0.9808*t) - (0.01532*t^2) + (t^3)%7129 [ t=. y - 1600 }}

NB. between +1700 and +1800:
NB.   $\Delta T = 8.83 + 0.1603 * t - 0.0059285 * t^2 + 0.00013336 * t^3 - t^4 / 1174000$ ; where: t = y - 1700
ry=. ry , 1700 1800
t5=. {{ 8.83 + (0.1603*t) - (0.0059285*t^2) + (0.00013336*t^3) - (t^4)%1174000 [ t=. y - 1700 }}

NB. between +1800 and +1860:
NB.   $\Delta T = 13.72 - 0.332447 * t + 0.0068612 * t^2 + 0.0041116 * t^3 - 0.00037436 * t^4$ 
NB.      + 0.0000121272 * t^5 - 0.0000001699 * t^6 + 0.000000000875 * t^7; where: t = y - 1800
ry =. ry , 1800 1860

```

```
t6=. {{ 13.72 - (0.332447*t) + (0.0068612*t^2) + (0.0041116*t^3) - (0.00037436*t^4) + (0.0000121272*t^5) -
>..> (0.0000001699*t^6) + 0.000000000875*t^7 [ t=. y - 1800 ]}}
```

*NB. between 1860 and 1900:*

*NB.  $\Delta T = 7.62 + 0.5737 * t - 0.251754 * t^2 + 0.01680668 * t^3$*

*NB.  $- 0.0004473624 * t^4 + t^5 / 233174$ ; where:  $t = y - 1860$*

*ry=. ry , 1860 1900*

```
t7=. {{ 7.62 + (0.5737*t) - (0.251754*t^2) + (0.01680668*t^3) - (0.0004473624*t^4) + (t^5)%233174 [ t=. y
>..>- 1860 ]}}
```

*NB. between 1900 and 1920:*

*NB.  $\Delta T = -2.79 + 1.494119 * t - 0.0598939 * t^2 + 0.0061966 * t^3 - 0.000197 * t^4$ ; where:  $t = y - 1900$*

*ry=. ry , 1900 1920*

```
t8=. {{ -2.79 + (1.494119*t) - (0.0598939*t^2) + (0.0061966*t^3) - 0.000197*t^4 [ t=. y - 1900 ]}}
```

*NB. between 1920 and 1941:*

*NB.  $\Delta T = 21.20 + 0.84493*t - 0.076100 * t^2 + 0.0020936 * t^3$ ; where:  $t = y - 1920$*

*ry=. ry , 1920 1941*

```
t9=. {{ 21.20 + (0.84493*t) - (0.076100*t^2) + 0.0020936*t^3 [ t=. y - 1920 ]}}
```

*NB. between 1941 and 1961:*

*NB.  $\Delta T = 29.07 + 0.407*t - t^2/233 + t^3 / 2547$ ; where:  $t = y - 1950$*

*ry=. ry , 1941 1961*

```
t10=. {{ 29.07 + 0.407*t - ((t^2)%233) + (t^3)%2547 [ t=. y - 1950 ]}}
```

*NB. between 1961 and 1986:*

*NB.  $\Delta T = 45.45 + 1.067*t - t^2/260 - t^3 / 718$  ; where:  $t = y - 1975$*

```
ry=. ry , 1961 1986
t11=. {{ 45.45 + (1.067*t) - ((t^2)%260) - (t^3)%718 [ t=. y - 1975 ]}}
```

*NB. between 1986 and 2005:*

*NB.  $\Delta T = 63.86 + 0.3345 * t - 0.060374 * t^2 + 0.0017275 * t^3 + 0.000651814 * t^4$*   
 *$+ 0.00002373599 * t^5$ ; where:  $t = y - 2000$*

```
ry=. ry , 1986 2005
t12=. {{ 63.86 + (0.3345*t) - (0.060374*t^2) + (0.0017275*t^3) + (0.000651814*t^4) + 0.00002373599*t^5 [ t
>..>=. y - 2000 ]}}
```

*NB. between 2005 and 2050:*

*NB.  $\Delta T = 62.92 + 0.32217 * t + 0.005589 * t^2$ ; where:  $t = y - 2000$*

```
ry=. ry , 2005 2050
t13=. {{ 62.92 + (0.32217*t) + 0.005589*t^2 [ t=. y - 2000 ]}}
```

*NB. between 2050 and 2150:*

*NB.  $\Delta T = -20 + 32 * ((y-1820)/100)^2 - 0.5628 * (2150 - y)$*

```
ry=. ry , 2050 2150
t14=. {{ _20 + (32 * ((y-1820)%100)^2) - 0.5628 * 2150 - y ]}}
```

*NB. after 2150:*

*NB.  $\Delta T = -20 + 32 * u^2$ ; where:  $u = (y-1820)/100$*

```
ry=. ry , 2150 3000
t15=. {{ _20 + 32 * U^2 [ U=. (y-1820)%100 ]}}
```

*NB. NOTE: the t(i) verbs match the intervals*

```
ti=. (rb=. /:~ ~. ,ry) I. y
```

```
'year range _1999 to 3000 exceeded' assert -(0,#rb) e. ti
```

*NB. t(i) gerund*

```
tg=. t1`t2`t3`t4`t5`t6`t7`t8`t9`t10`t11`t12`t13`t14`t15
```

*NB. apply t(i) verbs to appropriate intervals*

```
(;ti </. i.#y) { ;(tg {~ <: ~.ti) apply&.> ti </. y  
)
```

*NB. delta  $\Delta T$  decimal year: deltaTdy 2023 3 12 ,. 1959 12 11*

```
deltaTdy=: (0 { ] ) + 12 %~ 0.5 -~ 1 { ]
```

*NB. degrees from radian*

```
dfr=: *&57.2957795130823229
```

*NB. day minutes from hour minute time: dmfrhm 6 51 ,: 20 39*

```
dmfrhm=: [: +/"1 [: ] 60 1 *"1 ]
```

*NB. degrees, minutes, seconds from decimal degrees - inverse (ddfrdms)*

```
dmsfrdd=: <. (,.) 60 60 #: 3600 * 1 | ,
```

```
fmt_today=: 3 : 0
```

*NB.\*fmt\_today v-- format today verbs result.*

*NB.*

```
NB. monad: cl =. fmt_today (bt ; cl ; fl)
NB.
NB.   fmt_today nav_today 0
NB.   fmt_today (location_yellowstone~ 1935 7 6) iau_today 0

'Rs lName sRs cParms'=. y

NB. calc parameters
hdr=. <;._1' Location Sunrise Sunset Mag-Lim Above-Horz Dusk-Min Julian ΔT Longitude Latitude Year Month
>...>Day.dd UTCz'
cParms=. ctl ": <(rjust lName , (":sRs) , ": ,. cParms) ,. ' ' ,. >hdr

NB. rise/set - sorted by transit time
Rs=. >&.> <"1 |: Rs
Rs=. (('5.1'&(8!:2)@,. ) &.> 2 { Rs) (2)} Rs
Rs=. ('3.0'&(8!:2)&.> 3 { Rs) (3)} Rs
Rs=. ctl ": Rs ,:~ <;._1' Name Designation Tr-Alt-Deg Tr-24-HrMin'

cParms,LF,Rs
)

NB. fractional centuries from epoch J2000 Meeus pg. 83: gT0jd julfrcal 1957 10 4.81
gT0jd=: 36525 %~ 2451545. -- ]

NB. fractional centuries from epoch J2000 Meeus pg. 83: gT0ymd 1957 10 4.81
gT0ymd=: 36525 %~ 2451545. -- julfrcal
```

*NB. hours, minutes from decimal seconds: hmfrds dsfrhms 20 27 43.23*

```
hmfrds=: [: 24 60&#: 60 %~ ]
```

```
iau_today=: 3 : 0
```

*NB.\*iau\_today v-- named IAU stars rising/setting today.*

*NB.*

```
NB. monad: (bt ; clLoc ; itSrs ; flParms) =. iau_today uuIgnore
```

*NB.*

```
NB. iau_today 0
```

*NB.*

```
NB. dyad: (bt ; clLoc ; itSrs ; flParms) =. blymd_LB_UO_LMAG_LHORZ_LOC iau_today uuIgnore
```

*NB.*

```
NB. 'Riseset Location sRs cParms'=. (location_yellowstone~ 1935 7 6) iau_today 0
```

```
jd=. julfrcal ymd=. 3 {. 6!:0 ''
```

```
(ymd;jd;OBSLOCATION;UTCOFFSET;LIMITMAG;LIMITHORZ;LOCATIONNAME;DARKTRS) iau_today y
```

```
:
```

*NB. date, julian, location, UTC timezone, magnitude, horizon, location, dusk mins*

```
'YMD JD LB UO LMAG LHORZ LOCNAME DARK'=. x
```

*NB. star data*

```
'IAU NAV'=. loadstars 0
```

```
({."1 NAV)=. {:"1 NAV [ ({."1 IAU)=. {:"1 IAU
```

*NB. brighter magnitude limit !(\*)=. Vmag IAU\_Name Designation*

```
'Rs lName sRs cParms'=. x today_calc (LMAG > Vmag) # IAU_Name
```

```
NB. include Designation names
Rs=. 1 0 2 3 {"1 Rs ,.~ (IAU_Name i. 0 {"1 Rs){Designation
Rs;lName;sRs;cParms
)

intr3p=: 4 : 0

NB.*intr3p v-- interpolate three values - meeus pg 25.
NB.
NB. dyad: fln intr3p fl
NB.
NB. NB. meeus pg. 24
NB. yi=. 0.884226 0.877366 0.870531
NB. 0.05 intr3p yi

NB. y = y2 + (n/2)(a + b nc)
NB. a b c are differences

'only 3 values' assert 3=#y

d=. 1 2{alldifs y
'a b'=. >0{d [ c=. ,/ >1{d
(1{y) + (x%2) * a + b + x*c
)

julfrcal=: 3 : 0
```



```
NB.*julfrcal v-- Julian dates from calendar dates.
NB.
NB. Astronomical Julian date. Similiar to (tojulian) but handles
NB. the fact that Julian days start at noon rather than midnight
NB. for calendar days.
NB.
NB. monad: fl =. julfrcal ilYYYYMMDD / ftYYYYMMDD
NB.
NB. julfrcal 2001 9 11
NB. julfrcal 1776 1941 1867 , 7 12 7 ,: 4 7 1
NB.
NB. NB. Meeus (Astronomical Algorithms) test cases (pg. 61)
NB. NB. NOTE: the fractional day representation of time
NB. 2436116.31 = julfrcal 1957 10 4.81 NB. 7.a Sputnik 1
NB. 1842713.0 = julfrcal 333 1 27.5 NB. 7.b
NB.
NB. NB. zero date is roughly the age of the oldest bristlecone pines (coincidence?)
NB. julfrcal -4711 10 29.5

NB. vector J
'y m d'=. y
g=. 1582 <: y
f=. (d - d1) - 0.5 [ d1=. <. d
j=. - <. 7 * 4 %~ <.y + 12 %~ m+9

s=. * m-9 [ a=. | m-9
j3=. <. y + s * <. a%7

NB. scalar BASIC
NB. INPUT "Y,M,D ";Y,M,D
NB. G=1: IF Y<1582 THEN G=0
NB. D1=INT(D): F=D-D1-0.5
NB. J=-INT(7*(INT((M+9)/12)+Y)/4)
NB. IF G=0 THEN 805
NB. S=SGN(M-9): A=ABS(M-9)
NB. J3=INT(Y+S*INT(A/7))
```

```

j3=. - <. 3r4 * >: <. j3 % 100      NB. J3=-INT((INT(J3/100)+1)*3/4)
j=. j + (<.275 * m%9) + d1 + g*j3    NB. 805 J=J+INT(275*M/9)+D1+G*J3
j=. j + 1721027 + (2*g) + 367*y      NB. J=J+1721027+2*G+367*Y
b=. f >: 0                          NB. IF F>=0 THEN 825
f=. f + b [ j=. j - b                NB. F=F+1: J=J-1
f + j
)

```

*NB. left justify table*

```
ljust=: ' '&$: :([] |."_1~ i."1&0@([] e. [])
```

```
loadstars=: 3 : 0
```

```

NB.*loadstars v-- loads riseset star data.
NB.
NB. monad: bLIAU_Nav =. loadstars uuIgnore
NB.
NB. loadstars 0
NB.
NB. dyad: bLIAU_Nav=. pa loadstars uuIgnore
NB.
NB. 0 loadstars 0 NB. files
NB. 1 loadstars 0 NB. JOD
NB.
NB. loadstars~ 0 NB. idiom files
NB. loadstars~ 1 NB. idiom JOD
NB.

```

*NB. 2 loadstars 0 NB. files - define columns*

```
0 loadstars y
:
'invalid option' assert x e. 0 1 2

if. x e. 0 2 do.
  NB. load star data from addon directory !(*)=. jpath
  paddon=. jpath '~addons/jacks/testdata/'
  ciau=. read paddon, 'iau_named_stars_2022.txt'
  cnavs=. read paddon, 'Navigation_Stars.txt'
elseif. x-:1 do.
  NB. load star data from JOD (futs) !(*)=. get od require
  rc=. od ;:'futs utils' [ 3 od '' [ require 'general/jod'
  ciau=. ;{: , > {: MACRO_ajod_ get 'iau_named_stars_2022_txt'
  cnavs=. ;{: , > {: MACRO_ajod_ get 'Navigation_Stars_txt'
end.

ciau=. cold_iau_named_stars parse_iau_named_stars ciau
cnavs=. parsetd cnavs -. CR
cnavs=. (0 { cnavs) ,. <"1 |: }. cnavs
'star column overlap' assert 0 = #(0 {"1 cnavs) ([ -. -. ) 0 {"1 ciau

NB. define columns - override mixed assignments (<:)=:
if. x-:2 do.
  (0 {"1 ciau)=: 1 {"1 ciau
  (0 {"1 cnavs)=: 1 {"1 cnavs
```

```
(<ciau),(<cnavs),<(0 {"1 ciau),0 {"1 cnavs
else.
  (<ciau),<cnavs
end.
)
```

```
localsun=: 3 : 0
```

```
NB.*localsun v-- location sun rise/set times in hour minutes.
```

```
NB.
```

```
NB. monad:  itRs =. localsun blLB_UO_YMD
```

```
NB.
```

```
NB.    localsun OBSLOCATION;UTCOFFSET;6!:0 ''
```

```
'LB UO YMD'=. y
```

```
_2 ]\ ,sunriset1 (|.LB),UO,1 |. 3 {. YMD
```

```
)
```

```
location_home=: 3 : 0
```

```
NB.*location_home v-- set parameters for "home" location.
```

```
NB.
```

```
NB. monad:  bl =. location_home uuIgnore
```

```
NB.
```

```
NB.    location_home 0
```

```
NB.    NB. uses location with current date
```

```
NB.    fmt_today iau_today 0
```

```
NB.
NB. dyad:  bl =. flymfd location_home uuIgnore
NB.
NB.  NB. uses location with home date
NB.  (location_home 0) iau_today 0
NB.  (location_home 0) nav_today 0
NB.
NB.  NB. arbitrary dates for location
NB.  fmt_today (1712 3 15.34 location_home 0) nav_today 0
NB.  fmt_today (location_home~ 1933 9 25.75) iau_today 0

NB. test date https://www.almanac.com/astronomy/bright-stars/zipcode/83646/2023-03-27
2023 3 27 location_home y
:
JULIAN_riseset_=:  julfrcal ymd=. x

NB. longitude, latitude with standard signs
OBSLOCATION_riseset_=:  _116.375956 43.646775

LOCATIONNAME_riseset_=:  'Home - Meridian'

UTCOFFSET_riseset_=:  6.0    NB. MST time zone
LIMITMAG_riseset_=:  3.0    NB. stellar magnitude
LIMITHORZ_riseset_=:  20    NB. degrees above horizon
DARKTRS_riseset_=:  60      NB. minutes before and after sunset (0=ignore sun)

ymd;JULIAN;OBSLOCATION;UTCOFFSET;LIMITMAG;LIMITHORZ;LOCATIONNAME;DARKTRS
```

```
)

location_uluru=: 3 : 0

NB.*location_uluru v-- set parameters for Uluru location.
NB.
NB. monad: location_uluru uuIgnore
NB.
NB. location_uluru 0
NB. NB. uses location with current date
NB. iau_today 0
NB.
NB. dyad: bl =. flymfd location_uluru uuIgnore
NB.
NB. NB. uses location with uluru date
NB. (location_uluru 0) iau_today 0
NB.
NB. NB. arbitrary dates for location
NB. fmt_today (1712 3 15.34 location_uluru 0) nav_today 0
NB. fmt_today (location_uluru~ 1933 9 25.75) iau_today 0

2022 10 19 location_uluru y
:
JULIAN_riseset=: julfrcal ymd=. x

NB. longitude, latitude with standard signs
OBSLOCATION_riseset=: 131.01941 _25.34301
LOCATIONNAME_riseset=: 'Uluru - star party diner'
```

```
UTCOffset_riseset=: _9.5  NB. time zone
LIMITMAG_riseset=: 6.0    NB. stellar magnitude
LIMITHORZ_riseset=: 5     NB. degrees above horizon
DARKTRS_riseset=: 0       NB. minutes before and after sunset (0=ignore sun)
```

```
ymd;JULIAN;OBSLOCATION;UTCOffset;LIMITMAG;LIMITHORZ;LOCATIONNAME;DARKTRS
)
```

```
location_yellowstone=: 3 : 0
```

```
NB.*location_yellowstone v-- set parameters for Old Faithful location.
```

```
NB.
```

```
NB. monad: location_yellowstone uuIgnore
```

```
NB.
```

```
NB. location_yellowstone 0
```

```
NB. NB. uses location with current date
```

```
NB. iau_today 0
```

```
NB.
```

```
NB. dyad: bl =. flymfd location_yellowstone uuIgnore
```

```
NB.
```

```
NB. NB. uses location with yellowstone date
```

```
NB. (location_yellowstone 0) iau_today 0
```

```
NB.
```

```
NB. NB. arbitrary dates for location
```

```
NB. fmt_today (1712 3 15.34 location_yellowstone 0) nav_today 0
```

```
NB. fmt_today (location_yellowstone~ 1933 9 25.75) iau_today 0
```

```

2013 5 7 location_yellowstone y
:
JULIAN_riseset_=: julfrcal ymd=. x

NB. longitude, latitude with standard signs
OBSLOCATION_riseset_=: _110.82792 44.46057
LOCATIONNAME_riseset_=: 'Yellowstone - Old Faithful'

UTCOFFSET_riseset_=: 6.0    NB. MST time zone
LIMITMAG_riseset_=: 6.0    NB. stellar magnitude
LIMITHORZ_riseset_=: 10    NB. degrees above horizon
DARKTRS_riseset_=: 0       NB. minutes before and after sunset (0=ignore sun)

ymd;JULIAN;OBSLOCATION;UTCOFFSET;LIMITMAG;LIMITHORZ;LOCATIONNAME;DARKTRS
)

meanobliquityT0=: 3 : 0

NB.*meanobliquityT0 v-- mean obliquity of the ecliptic IAU in degrees.
NB.
NB. monad: fl =. meanobliquityT0 flT

NB. units are decimal arc seconds
ea=. +/3600 60 1 * OBLIQUITYDMS2000

NB. meeus (21.2) pg. 135
3600 %~ ea - (46.8150*y) - (0.00059*y^2) + 0.001813*y^3
)

```



```
meanobliquityT1=: 3 : 0
```

*NB.\*meanobliquityT1 v-- mean obliquity of the ecliptic Laskar in  
NB. degrees.*

*NB.*

*NB. Mean obliquity using Laskar's polynomial. This expression is  
NB. more accurate than (meanobliquityT0): see Meeus (21.2) pg.  
NB. 135.*

*NB.*

*NB. monad: fl =. meanobliquityT1 flT*

*NB. units are decimal arc seconds*

```
ea=. +/3600 60 1 * OBLIQUITYDMS2000
```

*NB. time units 10000 Julian years*

```
U=. y % 100
```

```
e0=. (39.05*U^6) + (7.12*U^7) + (27.87*U^8) + (5.79*U^9) + 2.45*U^10  
3600 %~ ea - (4680.93*U) - (1.55*U^2) + (1999.25*U^3) - (51.38*U^4) - (249.67*U^5) - e0  
)
```

```
meanobliquityjd0=: 3 : 0
```

*NB.\*meanobliquityjd0 v-- mean obliquity ecliptic for Julian date (y) degrees.*

*NB.*

*NB. monad: fl =. meanobliquityjd0 flJD*

*NB.*

*NB. NB. meeus pg. 136*

```
NB.  e0=. ,dmsfrdd meanobliquityjd0 2446895.5
NB.
NB.  NB. matches to 3 decimals
NB.  23 26 27.407 -: 0.001 round e0
NB.
NB. dyad: fl =. pa meanobliquityjd0 flJD
NB.
NB.  NB. Laskar algorithm
NB.  el=. ,dmsfrdd 1 meanobliquityjd0 2446895.5
```

```
0 meanobliquityjd0 y
:
meanobliquityT0`meanobliquityT1@.(x) gT0jd y
)
```

```
meansid0=: 4 : 0
```

```
NB.*meansid0 v-- mean sidereal time at Greenwich for T (x) JD (y).
NB.
NB. dyad: flDegr =. flT meansid flJD
```

```
NB. meeus (11.4) pg 84
280.46061837 + (360.98564736629 * y - 2451545.0) + (0.000387933 * x^2) - 38710000 %~ x^3
)
```

```
meansidjd0=: 3 : 0
```

*NB.\*meansidjd0 v-- mean sidereal time at Greenwich for julian day (y) in degrees.*

*NB.*

*NB. monad: fl =. meansidjd0 flJD*

*NB.*

*NB. NB. julian day for April 10, 1987 19h:24m:00s UT*

*NB. JD=. julfrcal 1987 4,10 + fdfrhms 19 21 0*

*NB. meansidjd0 JD*

*(gT0jd y) meansid0 y*

*)*

*nav\_today=: 3 : 0*

*NB.\*nav\_today v-- named navigation stars rising/setting today.*

*NB.*

*NB. monad: (bt ; clLoc ; itSrs ; flParms) =. nav\_today uuIgnore*

*NB.*

*NB. nav\_today 0*

*NB.*

*NB. dyad: (bt ; clLoc ; itSrs; flParms) =. blymd\_LB\_UO\_LMAG\_LHORZ\_LOC nav\_today uuIgnore*

*NB.*

*NB. 'Riseset Location sRs cParms'=. (location\_yellowstone~ 1935 7 6) nav\_today 0*

*jd=. julfrcal ymd=. 3 {. 6!:0 ''*

*(ymd;jd;OBSLOCATION;UTCOFFSET;LIMITMAG;LIMITHORZ;LOCATIONNAME;DARKTRS) nav\_today y*

*:*

*NB. star data*

*'IAU NAV'=. loadstars 0*

```
({"1 NAV)=. {"1 NAV [ ({"1 IAU)=. {"1 IAU
```

```
NB. !(*)=. Nav_Star_Name IAU_Name Designation  
'Rs lName sRs cParms'=. x today_calc Nav_Star_Name
```

```
NB. include Designation names  
Rs=. 1 0 2 3 {"1 Rs ,.~ (IAU_Name i. 0 {"1 Rs){Designation  
Rs;lName;sRs;cParms  
)
```

```
navdaylist=: 3 : 0
```

```
NB.*navdaylist v-- sky safari 6_0 observing list of today's navigation stars.
```

```
NB.
```

```
NB. The files created by this verb can be loaded into the Sky
```

```
NB. Safari iOS and Mac apps.
```

```
NB.
```

```
NB. monad: cl =. navdaylist uuIgnore
```

```
NB.
```

```
NB. navhome=. navdaylist 0
```

```
NB. navhome write jpath '~JODIMEX/Navigation_Stars_Home.skylist'
```

```
NB. j profile !(*)=. jpath
```

```
skl=. read jpath '~addons/jacks/testdata/Navigation_Stars.skylist'
```

```
'st loc cParms'=. nav_today 0 [ location_home 0
```

```
NB. skylist header
```

```
cst=. 'SortedBy=Default Order'
```

```
hdr=. cst ((,&LF)@[ ,~ beforestr) skl
```

*NB. cut skylist objects*

```
sob=. (] <;.1~ 'SkyObject=BeginObject' E. ]) cst afterstr skl
```

*NB. retain objects that match star and hdr names*

```
b=. +./ (0 {"1 st) +./@E.&>"0 1 sob
```

```
sob=. sob #~ b *. +./ (1 {"1 st) +./@E.&>"0 1 sob
```

*NB. reset sort order*

```
sob=. ];._2 tlf ;sob
```

```
ix=. I. +./"1 (,:'DefaultIndex=') E. sob
```

```
ns=. '='&beforestr"1 ix{sob
```

```
ns=. ns ,. '=' ,. ljust ": ,. i. #ns
```

```
hdr,ctl >(<"1 ns) (ix)} <"1 sob
```

```
)
```

*NB. normalize negative degree sidereal time: nnth0 -1677831.2621266*

```
nnth0=: ] + 360 * [: | [: (<.) 360 %~ ]
```

*NB. normalize positive degree sidereal time: npth0 1677831.2621266*

```
npth0=: ] - 360 * [: (<.) 360 %~ ]
```

*NB. normalize degree sidereal time: nth0 \_35555 77777*

```
nth0=: npth0`nnth0@.(0&>:@[])
```

```
nutaton_longitude_dPsi=: 3 : 0
```

```
NB.*nututation_longitude_dPsi v-- nututation in ecliptical longitude in degrees (1980 iau theory).
NB.
NB. NOTE: the pseudo-code is vector ready and easily converted to J.
NB.
NB. verbatim: algorithm from Jay Tanner https://neoprogrammics.com/nutations/
NB.
NB. see: nututation_in_longitude_dPsi_md
NB.
NB. monad: flDeg = . nututation_longitude_dPsi flJD
NB.
NB. ymd=. |: 2023 3 12 , 1959 12 11 , 2135 12 13, 1700 1 1 ,: 1935 7 6
NB. JD=. julfrcal ymd NB. no delT adj.
NB. 2460015.5 = 0{JD
NB. nututation_longitude_dPsi JD
NB.
NB. NB. see (futs) test: (riseset_tanner_smoke) for examples

T=. (y - 2451545) % 36525 NB. T = (JD - 2451545) / 36525
T2=. T*T NB. T2 = T*T
T3=. T*T2 NB. T3 = T*T2

NB. DegToRad = 3.1415926535897932 / 180
DegToRad=. 3.1415926535897932 % 180

NB. w1 = 297.85036 + 445267.11148*T - 0.0019142*T2 + (T3 / 189474)
w1=. 297.85036 + (445267.11148*T) - (0.0019142*T2) + (T3 % 189474)
w1=. DegToRad*(w1) NB. w1 = DegToRad*(w1)
```

```

NB. w2 = 357.52772 + 35999.05034*T - 0.0001603*T2 - (T3 / 300000)
w2=. 357.52772 + (35999.05034*T) - (0.0001603*T2) - (T3 % 300000)
w2=. DegToRad*(w2)          NB. w2 = DegToRad*(w2)

NB. w3 = 134.96298 + 477198.867398*T + 0.0086972*T2 + (T3 / 56250)
w3=. 134.96298 + (477198.867398*T) + (0.0086972*T2) + (T3 % 56250)
w3=. DegToRad*(w3)          NB. w3 = DegToRad*(w3)

NB. w4 = 93.27191 + 483202.017538*T - 0.0036825*T2 + (T3 / 327270)
w4=. 93.27191 + (483202.017538*T) - (0.0036825*T2) + (T3 % 327270)
w4=. DegToRad*(w4)          NB. w4 = DegToRad*(w4)

NB. w5 = 125.04452 - 1934.136261*T + 0.0020708*T2 + (T3 / 450000)
w5=. 125.04452 - (1934.136261*T) + (0.0020708*T2) + (T3 % 450000)
w5=. DegToRad*(w5)          NB. w5 = DegToRad*(w5)

w=. (sin w5)*((_174.2*T) - 171996)          NB. w = sin(w5)*(-174.2*T - 171996)
w=. w + (sin 2 * w4 + w5 - w1)*((_1.6*T) - 13187)  NB. w = w + sin(2*(w4 + w5 - w1))*(-1.6*T - 13187)
w=. w + (sin 2 * w4 + w5)*(_2274 - 0.2*T)          NB. w = w + sin(2*(w4 + w5))*(-2274 - 0.2*T)
w=. w + (sin 2 * w5)*((0.2*T) + 2062)              NB. w = w + sin(2 * w5)*(0.2*T + 2062)
w=. w + (sin w2)*(1426 - 3.4*T)                   NB. w = w + sin(w2)*(1426 - 3.4*T)
w=. w + (sin w3)*((0.1*T) + 712)                  NB. w = w + sin(w3)*(0.1*T + 712)

NB. w = w + sin(2*(w4 + w5 - w1) + w2)*(1.2*T - 517)
w=. w + (sin (2 * w4 + w5 - w1) + w2)*((1.2*T) - 517)

```

```

w=. w + (sin (2*w4) + w5)*((_0.4*T) - 386)      NB. w = w + sin(2 * w4 + w5)*(-0.4*T - 386)

NB. w = w + sin(2*(w4 + w5 - w1) - w2)*(217 - 0.5*T)
w=. w + (sin (2 * w4 + w5 - w1) - w2)*(217 - 0.5*T)

w=. w + (sin (2*w4 - w1) + w5)*(129 + 0.1*T)      NB. w = w + sin(2*(w4 - w1) + w5)*(129 + 0.1*T)
w=. w + (sin w3 + w5)*((0.1*T) + 63)              NB. w = w + sin(w3 + w5)*(0.1*T + 63)
w=. w + (sin w5 - w3)*((_0.1*T) - 58)            NB. w = w + sin(w5 - w3)*(-0.1*T - 58)
w=. w + (sin 2*w2)*(17 - 0.1*T)                  NB. w = w + sin(2*w2)*(17 - 0.1*T)
w=. w + (sin 2 * w2 + w4 + w5 - w1)*((0.1*T) - 16) NB. w = w + sin(2*(w2 + w4 + w5 - w1))*((0.1*T) - 16)
w=. w - 301*(sin (2 * w4 + w5) + w3)              NB. w = w - 301*sin(2*(w4 + w5) + w3)
w=. w - 158*(sin w3 - 2*w1)                      NB. w = w - 158*sin(w3 - 2*w1)
w=. w + 123*(sin (2 * w4 + w5) - w3)              NB. w = w + 123*sin(2*(w4 + w5) - w3)
w=. w + 63*(sin 2*w1)                            NB. w = w + 63*sin(2*w1)
w=. w - 59*(sin (2 * w1 + w4 + w5) - w3)          NB. w = w - 59*sin(2*(w1 + w4 + w5) - w3)
w=. w - 51*(sin (2*w4) + w3 + w5)                NB. w = w - 51*sin(2 * w4 + w3 + w5)
w=. w + 48*sin(2 * w3 - w1)                      NB. w = w + 48*sin(2*(w3 - w1))
w=. w + 46*(sin (2 * w4 - w3) + w5)              NB. w = w + 46*sin(2*(w4 - w3) + w5)
w=. w - 38*(sin 2 * w1 + w4 + w5)                NB. w = w - 38*sin(2*(w1 + w4 + w5))
w=. w - 31*(sin 2 * w3 + w4 + w5)                NB. w = w - 31*sin(2*(w3 + w4 + w5))
w=. w + 29*(sin 2*w3)                            NB. w = w + 29*sin(2*w3)
w=. w + 29*(sin (2 * w4 + w5 - w1) + w3)          NB. w = w + 29*sin(2*(w4 + w5 - w1) + w3)
w=. w + 26*(sin 2*w4)                            NB. w = w + 26*sin(2*w4)
w=. w - 22*(sin 2* w4 - w1)                      NB. w = w - 22*sin(2*(w4 - w1))
w=. w + 21*(sin (2*w4) + w5 - w3)                NB. w = w + 21*sin(2*w4 + w5 - w3)
w=. w + 16*(sin (2*w1) - w3 + w5)                NB. w = w + 16*sin(2*w1 - w3 + w5)
w=. w - 15*(sin w2 + w5)                        NB. w = w - 15*sin(w2 + w5)

```



```

w=. w - 13*(sin w3 + w5 - 2*w1)
w=. w - 12*(sin w5 - w2)
w=. w + 11*(sin 2 * w3 - w4)
w=. w - 10*(sin (2 * w4 + w1) + w5 - w3)
w=. w - 8*(sin (2 * w4 + w1 + w5) + w3)
w=. w + 7*(sin (2 * w4 + w5) + w2)
w=. w - 7*(sin w3 - (2*w1) + w2)
w=. w - 7*(sin (2 * w4 + w5) - w2)
w=. w - 7*(sin (2*w1) + (2*w4) + w5)
w=. w + 6*(sin (2*w1) + w3)
w=. w + 6*(sin 2 * w3 + w4 + w5 - w1)
w=. w + 6*(sin (2 * w4 - w1) + w3 + w5)
w=. w - 6*(sin (2 * w1 - w3) + w5)
w=. w - 6*(sin (2*w1) + w5)
w=. w + 5*(sin w3 - w2)
w=. w - 5*(sin (2* w4 - w1) + w5 - w2)
w=. w - 5*(sin w5 - 2*w1)
w=. w - 5*(sin (2 * w3 + w4) + w5)
w=. w + 4*(sin (2 * w3 - w1) + w5)
w=. w + 4*(sin (2 * w4 - w1) + w2 + w5)
w=. w + 4*(sin w3 - 2*w4)
w=. w - 4*(sin w3 - w1)
w=. w - 4*(sin w2 - 2*w1)
w=. w - 4*(sin w1)
w=. w + 3*(sin (2*w4) + w3)
w=. w - 3*(sin 2 * w4 + w5 - w3)
w=. w - 3*(sin w3 - w1 - w2)

```

```

NB. w = w - 13*sin(w3 + w5 - 2*w1)
NB. w = w - 12*sin(w5 - w2)
NB. w = w + 11*sin(2*(w3 - w4))
NB. w = w - 10*sin(2*(w4 + w1) + w5 - w3)
NB. w = w - 8*sin(2*(w4 + w1 + w5) + w3)
NB. w = w + 7*sin(2*(w4 + w5) + w2)
NB. w = w - 7*sin(w3 - 2*w1 + w2)
NB. w = w - 7*sin(2*(w4 + w5) - w2)
NB. w = w - 7*sin(2*w1 + 2*w4 + w5)
NB. w = w + 6*sin(2*w1 + w3)
NB. w = w + 6*sin(2*(w3 + w4 + w5 - w1))
NB. w = w + 6*sin(2*(w4 - w1) + w3 + w5)
NB. w = w - 6*sin(2*(w1 - w3) + w5)
NB. w = w - 6*sin(2*w1 + w5)
NB. w = w + 5*sin(w3 - w2)
NB. w = w - 5*sin(2*(w4 - w1) + w5 - w2)
NB. w = w - 5*sin(w5 - 2*w1)
NB. w = w - 5*sin(2*(w3 + w4) + w5)
NB. w = w + 4*sin(2*(w3 - w1) + w5)
NB. w = w + 4*sin(2*(w4 - w1) + w2 + w5)
NB. w = w + 4*sin(w3 - 2*w4)
NB. w = w - 4*sin(w3 - w1)
NB. w = w - 4*sin(w2 - 2*w1)
NB. w = w - 4*sin(w1)
NB. w = w + 3*sin(2*w4 + w3)
NB. w = w - 3*sin(2*(w4 + w5 - w3))
NB. w = w - 3*sin(w3 - w1 - w2)

```

---

```

w=. w - 3*(sin w2 + w3)
w=. w - 3*(sin (2 * w4 + w5) + w3 - w2)
w=. w - 3*(sin (2 * w1 + w4 + w5) - w2 - w3)
w=. w - 3*(sin (2 * w4 + w5) + 3*w3)
w=. w - 3*(sin (2 * w1 + w4 + w5) - w2)

NB. w = w - 3*sin(w2 + w3)
NB. w = w - 3*sin(2*(w4 + w5) + w3 - w2)
NB. w = w - 3*sin(2*(w1 + w4 + w5) - w2 - w3)
NB. w = w - 3*sin(2*(w4 + w5) + 3*w3)
NB. w = w - 3*sin(2*(w1 + w4 + w5) - w2)

dPsiDeg=. w % 36000000.0 NB. dPsiDeg = w / 36000000.0
)

parse_iau_named_stars=: 3 : 0

NB.*parse_iau_named_stars v-- IAU named star list to btcl header
NB. table.
NB.
NB. Original star name data was downloaded from:
NB.
NB. https://www.iau.org/public/themes/naming\_stars/
NB.
NB. and slightly adjusted in Excel and saved as a Unicode UTF-8
NB. CSV export.
NB.
NB. monad: btcl =. parse_iau_named_stars clTxt
NB.
NB. NB. get stars
NB. iau=. read jpath '~addons/jacks/testdata/iau_named_stars_2022.txt'
NB. parse_iau_named_stars iau

NB. parse utf8 csv

```

```
t=. parsebomcsv y

NB. extract relevant columns
c=. ;:'IAU_Name Designation HIP Bayer_Name Vmag RA_J2000 Dec_J2000'
t=. t {"1~ (0 { t) i. c

NB. scrub objects with questionable magnitude
t #~ _ ~: _999&".&> (c i. <'Vmag') {"1 t
)

NB. parses utf8 csv files with optional BOM mark
parsebomcsv=: [: parsecsv [: utf8 ] }~ 0 3 { ~ (239 187 191{a.) -: 3 {. ]

parsecsv=: 3 : 0

NB.*parsecsv v-- parses comma delimited files. (x) is the field
NB. delimiter. Lines are delimited with either CRLF or LF
NB.
NB. monad: btcl =. parsecsv cl
NB. dyad: btcl =. ca parsecsv cl
NB.
NB. ', ' parsecsv read 'c:\comma\delimited\text.csv'

', ' parsecsv y
:
'separator cannot be the " character' assert -. x -: ''
```

*NB. CRLF delimited \*.csv text to char table*

```
y=. x ,. ] ;. _2 y -. CR
```

*NB. bit mask of unquoted " field delimiters*

```
b=. -. }. ~:/\ '"" e.~ ' ' , , y
```

```
b=. ($y) $ b *. , x = y
```

*NB. use masks to cut lines*

```
b <;. _1"1 y  
)
```

*NB. parse TAB delimited table text - see long document*

```
parsetd=: [: <;. _2&> (a.{~9) ,&.>~ [: <;. _2 [: (] , ((10{a.})"_ = {:) }. (10{a.})"_ ) (13{a.}) -.~ ]
```

*NB. reads a file as a list of bytes*

```
read=: 1!:1&([`<@.(32&>@{3!:0)))
```

*NB. radians from degrees*

```
rfd=: *&0.0174532925199432955
```

```
riseset=: 4 : 0
```

*NB.\*riseset v-- rise, transit, set times of IAU named stars.*

*NB.*

*NB. dyad: (btRs ; flParms) =. bLYMD\_UO\_LB\_AOBJ riseset blclStarNames*

*NB.*

---

```

NB.    LB=.  _116.375956 43.646775    NB. Meridian
NB.    YMD=. 2023 3 27
NB.    UO=. 6
NB.    (YMD;UO;LB) riseset 'Algol'
NB.    (YMD;UO;LB) riseset 'Algol';'Rigel';'Spica'
NB.
NB.    NB. add objects not IAU names - need name, ra, dec
NB.    AOB=. (<;:'Venus'),(<41.73129),<18.44092
NB.    AOB=. ,&.> (;:'OBJ_Name OBJ_RA_J2000 OBJ_Dec_J2000') ,. AOB
NB.    (YMD;UO;LB;<AOB) riseset 'Venus'

NB. local time, UT offset (0=Greenwich), Latitude Longitude
'ymfd uo LB AOB'=. 4 {. x

NB. convert LB to meeus convention
LB=. _1 1 * LB

NB. local time to UT
UT=. ymfd + 0 0,uo%24

NB. look up RA, Dec
'IAU Navigation'=. loadstars 0
NB. IAU stars !(*)=. IAU_Name RA_J2000 Dec_J2000
({."1 IAU)=. {:"1 IAU
Stars=. boxopen y

if. #AOB do.

```

---

```

NB. insert additional objects
({."1 AOB)=. {:"1 AOB
NB. !(*)=. OBJ_Dec_J2000 OBJ_Name OBJ_RA_J2000
IAU_Name=. OBJ_Name , IAU_Name
RA_J2000=. OBJ_RA_J2000 , RA_J2000
Dec_J2000=. OBJ_Dec_J2000 , Dec_J2000
end.

if. 0 e. b=. Stars e. IAU_Name do.
  smoutput 'not in IAU named stars -> '; Stars #~ -.b
else.
  ix=. IAU_Name i. Stars
  RA=. <ix{RA_J2000 [ Dec=. <ix{Dec_J2000
  riseset_calc UT;uo;LB;(<Stars),RA,Dec
end.
)

riseset_calc=: 3 : 0

NB.*riseset_calc v-- rise, transit, set times of stars.
NB.
NB. Main rise/set calculations. Argument (y) set in (riseset).
NB.
NB. monad: (btRs ; flParms) =. riseset_calc bLYMD_UO_LB_OBJ_RA_Dec

'ymd uo LB obj ra dec'=. ,&.> y

NB. (L) longitude, west positive

```

*NB. (B) latitude, north positive*

*'L B'=. LB*

*obj=. obj , "0 1 a:,a: NB. result table*

*NB. dynamical time  $\Delta T$  in fractional days NOTE:  $\Delta T$  is not*

*NB. going to change a lot over the interpolation period !(\*)=. nc*

*if. 0=nc<'DeltaTsOverride\_risese' do. dTs=. DeltaTsOverride\_risese\_*  
*else.*

*dTs=. ./deltaT0 deltaTdy ymd*

*end.*

*dTfd=. dTs%DAYSECS*

*NB. apparent sidereal time Greenwich at 0h in degrees*

*th0=. ./ddfrdms 15 \* apparsidjd0 JD=. julfrcal ymd*

*NB. TD times  $\Delta T + UT = TD$*

*TD=. (2 {. ymd), "1 0 (\_1 0 1 + {:ymd) + dTfd*

*NB. apparent ra,dec for \_1 0 1 days around rise/set*

*rdi=. |: TD apparRADEC"1 \_ ra ,: dec*

*h0=. STDALTITUDE*

*NB. approximate times (14.1) meeus pg. 98*

*cosH0=. ((sind h0) - (sind B)\*sind (<a::;1;1){rdi) % (cosd B)\*cosd (<a::;1;1){rdi*

*NB. 1 indicates above or below horizon*

```
bhrz=. 1 < |cosH0
obj=. (<"0 bhrz) (<a;;1)} obj
obj=. (<'above or below horizon') (<(I. bhrz);2)} obj
ix=. I. -.bhrz  NB. objects that rise and set

NB. m(i) are fractional day times (1/) puts mi in [0,1]
H0=. dfr arccos ix{cosH0
m0=. 1|360 %~ ((<ix;0;1){rdi) + L - th0
m1=. 1|m0 - H0 % 360
m2=. 1|m0 + H0 % 360

NB. rise, transit, setting
m=. m1 ,. m0 ,. m2

NB. sidereal time at Greenwich - meeus pg. 99
th=. nth0 th0 + 360.985647*m

NB. adjusted ra,dec
rda=. nu intr3p"1 ix{rdi [ nu=. dTfd + m

NB. local hour angles
rax=. <a;;0 [ decx=. <a;;1
H=. (th - L) - rax{rda

NB. body's altitude (12.6) meeus pg. 89
sih=. ((sind B)*sind decx{rda) + (cosd B)*(cosd decx{rda)*cosd H
```



*NB. degree altitudes positive*

```
h=. |dfr arcsin sih
```

*NB. corrections for transits (trx), rise/sets (rsx)*

```
dltm=. ($m)$0
trx=. <a;;1 [ rsx=. <a;;0 2
dltm=. (-(trx{H}%360) trx} dltm
drs=. rsx { (h - h0) % 360 * (cosd decx{rda)*(cosd B)*sind H
dltm=. drs rsx} dltm
m=. m + dltm
```

*NB. calc parameters Julian date, ΔT, Longitude, Latitude, ymfd, timez*

```
cParms=. JD,dTs,(-L),B,ymd,uo
```

*NB. objects, above/below, altitudes, fractional day UT, UT hours/minutes*

```
cParms ;~ (<"2 (,."1 ] 0.5 round h) ,"1 (,."1 m) ,"1 ] 1 round hmfrds DAYSECS*m) (<ix;2)} obj
)
```

*NB. right justify table*

```
rjust=: ' '&$: :([ |."_1~ +/"1@(-.@(<./\"1@([ = ]))))
```

*NB. round (y) to nearest (x) (e.g. 1000 round 12345)*

```
round=: [ * [: (<.) 0.5 + %~
```

*NB. sine radians*

```
sin=: 1&o.
```

*NB. sin degrees*

sind=: sin@rfd

*NB. session manager output*

smoutput=: 0 0 \$ 1!:2&2

sunriseset1=: 3 : 0

*NB.\*sunriseset1 v-- computes sun rise and set times - see group*

*NB. documentation.*

*NB.*

*NB. This verb has been adapted from a BASIC program submitted by*

*NB. James Brimhall to \*Sky & Telescope's\* "shortest sunrise/set*

*NB. program" contest. Winning entries were listed in the March*

*NB. 1995 Astronomical Computing column.*

*NB.*

*NB. monad: itHM =. sunriseset1 flBLHMDY / ftBHMDY*

*NB.*

*NB. NB. rise and set times observer location today*

*NB. td=. (/OBSLOCATION) , UTCOFFSET, 1 |. 3 {. 6!:0 ''*

*NB. sunriseset1 td*

*NB.*

*NB. NB. rise and set times on June 30 1995 on Greenwich meridian*

*NB. t0=. 0 0 0 6 30 1995 NB. equator*

*NB. t1=. 49 0 0 6 30 1995 NB. north - lat of western US/Canada border*

*NB. t2=. \_47 0 0 6 30 1995 NB. south - southern Chile and Argentina*

*NB. t3=. 75 0 0 6 30 1995 NB. far north (sun always up)*

*NB. t4=. \_75 0 0 6 30 1995 NB. far south (sun always down)*

*NB. latitude, longitude, time-zone, month, day, year !(\*)=. la lo tz m d y*  
y=. # la [ 'la lo tz m d y'=. |: tabit y  
dr=. 1r180p1 [ dd=. 360 % 365.25636 [ rt=. 50r60

*NB. days into year with leap year adjustment*  
dm=. 0 31 59 90 120 151 181 212 243 273 304 334  
dl=. (2 {. dm) , >: 2 }. dm  
bl=. 0 = 4 | y [ m=. <: m  
dy=. d + ((-.bl) \* m { dm) + bl \* m { dl  
dy=. 0.5 + dy - lo % 360

*NB. (th) angle Earth has moved since winter solstice*  
th=. 9.357001 + (dd \* dy) + 1.914 \* sin dr \* (dd \* dy) - 3.97  
c3=. 0.3978 \* cos dr \* th  
dc=. (- % dr) \* arctan c3 % %: 1 - c3 ^ 2

*NB. adjust for positive and negative latitudes*  
bl=. la < 0  
a1=. ((-.bl) \* (90 - la) + dc) + bl \* (90 + la) - dc  
a2=. ((-.bl) \* (la - 90) + dc) + bl \* (\_90 - la) - dc

*NB. sun never rises or sets masks*  
nvset =. a2 >: - rt [ nvrise=. a1 < - rt

*NB. corrections*

```

drla=. dr * la [ drdc=. dr * dc
c1=. ((sin - dr * rt) - (sin drdc) * sin drla) % (cos drdc) * cos drla
t2=. dr %~ arctan (:%: 1 - c1 ^ 2) % c1
t1=. 360 - t2 [ bl=. c1 < 0
t2=. (t2 * -.bl) + bl * 180 + t2
t1=. (t1 * -.bl) + bl * 360 - t2

```

*NB. first order equation of time*

```

et=. 0.1511 * sin dr * 17.86 + 2 * dddy=. dd * dy
et=. (_0.1276 * sin dr * dddy - 3.97) - et
drla=. drdc=. dddy=. 0

```

*NB. time zone adjusted rise and set times*

```

tr=. (t1 % 15) - 12 [ ts=. t2 % 15
tr=. tr - et [ ts=. ts - et
s=. ts + tc [ r=.tr + tc [ tc=(-tz) - lo % 15
hrmn=. (<. r) ,: 1 round 60 * 1|r
hrmn=. hrmn , (<.12 + s) ,: 1 round 60 * 1|s

```

*NB. adjust for when sun never rises or sets*

```

hrmn=. hrmn *"1 -. bl [ bl=. nvset +. nvrise
hrmn=. NORISESET (<0;bl # pos) } hrmn [ pos=. i. {: $ hrmn
1 (<1;nvset # pos) } hrmn
)

```

*NB. promotes only atoms and lists to tables*

```

tabit=: ]`,:@.(1&>:@(#@$))^:2

```

*NB. appends trailing line feed character if necessary*

```
tlf=: ] , ((10{a.)"_ = {:) }. (10{a.)"_
```

```
today_calc=: 4 : 0
```

*NB.\*today\_calc v-- named (y) stars rising/setting today.*

*NB.*

```
NB. dyad: (bt ; clLoc ; itSrs ; flParms) =. bl today_calc blclIauStars
```

*NB.*

```
NB. IauStars=. ;:'Algol Rigel Spica'
```

```
NB. 'Riseset lName sRs cParms'=. (location_uluru 0) today_calc IauStars
```

*NB. date, julian, lat/lon, UTCz, magnitude, horizon, location, dusk minutes*

```
'YMD JD LB UO LMAG LHORZ LOCNAME DARK'=. x
```

```
'Rsiau cParms'=. (YMD;UO;LB) riseset y [ srs=. localsun LB;UO;YMD
```

*NB. retain rising setting - circumpolar NIMP*

```
Rsiau=. Rsiau #~ -. ; 1 {"1 Rsiau
```

*NB. name ,. transit altitude, hour minutes*

```
ahm=. 1&{&.> 2 {"1 Rsiau
```

```
Rsiau=. (0 {"1 Rsiau) ,. (0 {&.> ahm) ,. (<2 3){&.> ahm
```

*NB. retain above local horizon*

```
Rsiau=. Rsiau #~ LHORZ < 0&{&> 1 {"1 Rsiau
```

*NB. retain stars transiting when dark*

```
if. 0<DARK do. Rsiau=. Rsiau #~ (>{: "1 Rsiau) darktransits srs;DARK end.
```

*NB. sort by transit time*

```
(LOCNAME;srs;LMAG,LHORZ,DARK,cParms) ;~ Rsiau {~ /: >2 {"1 Rsiau  
)
```

*NB. character list to UTF-8*

```
utf8=: 8&u:
```

```
zetzthT0=: 3 : 0
```

*NB.\*zetzthT0 v-- epoch adjustment terms for J2000 RA DEC in degrees.*

*NB.*

*NB. monad: fT =. zetzthT0 ftYYYYMMDD*

*NB.*

*NB. zetzthT0 2028 11 13.19*

*NB.*

*NB. zetzthT0 2023 4 23 , 1988 3 20 ,: 1987 4,10 + fdfrhms 19 21 0*

```
t=. gT0ymd y
```

```
't2 t3'=. t (^"1 0) 2 3 NB. t~2 and t~3
```

*NB. meeus (20.3) pg. 126*

```
zet=. (2306.2181*t) + (0.30188*t2) + 0.017988*t3
```

```
z=. (2306.2181*t) + (1.09468*t2) + 0.018203*t3
```

```
th=. (2004.3109*t) + (0.42665*t2) + 0.041833*t3
```

*NB. insure degree result rank matches (y) rank*

```
3600 %~ zet , z (,`,: )@.(2=#$y) th
)
```

*NB.POST\_riseset post processor.*

```
smoutput IFACE=: (0 : 0)
```

```
NB. (riseset) interface word(s): 20230519j104416
```

```
NB. -----
```

```
NB. baby_today  NB. named Babylonian stars rising/setting today
```

```
NB. fmt_today   NB. format today verbs result
```

```
NB. iau_today   NB. named IAU stars rising/setting today
```

```
NB. loadstars   NB. loads riseset star data
```

```
NB. nav_today   NB. named navigation stars rising/setting today
```

```
NB. navdaylist  NB. sky safari 6_0 observing list of today's navigation stars
```

```
NB. riseset     NB. rise, transit, set times of stars
```

```
    fmt_today nav_today location_yellowstone 0
```

```
)
```

```
cocurrent 'base'
```

```
coinset   'riseset'
```

# Index

(...)=:, 27

afterstr, 9

alldifs, 9

apparRADEC, 9

apparsecs, 10

apparsidjd0, 10

apply, 10

arccos, 10

arcsin, 10

arctan, 11

assert, 11

atan2, 11

baby\_today, 13

babylonian\_named\_stars, 13

beforestr, 14

boxopen, 14

cold\_iau\_named\_stars, 15

cos, 15

cosd, 15

CR, 7

ctl, 15

darktransits, 16

DARKTRS, 7

DARKTRS\_riseset\_, 29, 31, 32

DAYSECS, 7

ddfrdms, 16

deltaT0, 17

deltaTdy, 21

dfr, 21

dmfrhm, 21

dmsfrdd, 21

fmt\_today, 21

gT0jd, 22

gT0ymd, 22

hmfrds, 23

iau\_today, 23

IFACE, 55

IFACEWORDSriseset, 7

intr3p, 24

julfrcal, 24

JULIAN, 7

JULIAN\_riseset\_, 29, 30, 32

LF, 7

LIMITHORZ, 7

LIMITHORZ\_riseset\_, 29, 31, 32

LIMITMAG, 7

LIMITMAG\_riseset\_, 29, 31, 32

ljust, 26

loadstars, 26

localsun, 28

location\_home, 28

location\_uluru, 30

location\_yellowstone, 31

LOCATIONNAME, 8

LOCATIONNAME\_riseset\_, 29, 30, 32

meanobliquityjd0, 33

meanobliquityT0, 32

meanobliquityT1, 33

meansid0, 34

meansidjd0, 34

nav\_today, 35

navdaylist, 36

nnth0, 37

NORISESET, 8

npth0, 37

nth0, 37

nutration\_longitude\_dPsi, 37

OBLIQUITYDMS2000, 8

OBSLOCATION, 8



OBSLOCATION\_risese, 29, 30, 32

parse\_iau\_named\_stars, 42

parsebomcsv, 43

parsecsv, 43

parsetd, 44

read, 44

rfd, 44

risese, 44

risese\_calc, 46

rjust, 49

ROOTWORDSrisese, 8

round, 49

sin, 49

sind, 50

smoutput, 50

STDALTITUDE, 8

sunrisese1, 50

tabit, 52

tlf, 53

today\_calc, 53

UTCOffset, 8

UTCOffset\_risese, 29, 31, 32

utf8, 54

VMDrisese, 8

zetzthT0, 54