```
! Fortran 90 Module
! Simplified BSD Licence (below). Enjoy!
Module standish
      Implicit None
! standish ephemeris
!  * see http://ssd.jpl.nasa.gov/txt/aprx_pos_planets.pdf
    ! elements
    ! a = semi major axis (au)
    ! e = eccentricity (rad)
    ! i = inclination (rad)
    ! l = mean longitude (rad)
    ! w = longitude of perihelion (rad)
    ! o = longitude of ascending mode (rad)
!
! global user defined type
      Type ephem
         Character (Len=64) :: desc ! data description
         Integer :: n           ! number of planets
         Logical :: lrad        ! .true. = table in radians
         Real (8) :: epoch      ! data epoch
         Real (8) :: jul1, jul2 ! valid date range
         Character (Len=8), Dimension (10) :: name ! planet name
         Real (8), Dimension (16, 9) :: o ! keplerian elements terms
      End Type ephem
!
! global variables
      Type (ephem) :: eph (2)! approximate keplerian elements
      Integer :: i, j !only needed initially for data statements
      Character (Len=64) :: SMODVER = "Standish Ephemeris Module 2018 V1"
      Real (8), Parameter :: s_ZERO = 0.0d00, s_ONE = 1.0d00, s_TWO = 2.0d00
      Real (8), Parameter :: s_D2PI = s_TWO * Acos (-s_ONE)! 2Pi
      Real (8), Parameter :: s_DR2D = 360.0d0/s_D2PI ! Rad to Deg
      Real (8), Parameter :: s_SOBL = 0.397776978d0  ! sin(23.43928 deg) J2000 Obliquity
      Real (8), Parameter :: s_COBL = 0.917482139d0  ! cos(23.43928 deg) J2000 Obliquity
      Real (8), Parameter :: s_KPS = 4.74047046d0    ! AU/YR -> km/s velocity conversion
      Real (8), Parameter :: s_DPC = 3.6525d04       ! Julian days per century
      Real (8), Parameter :: mu_sun = 39.47692641d0  ! AU^3/YR^2

! DATA
! Approximate Positions of the Major Planets -
    ! Data and Approximation Model from E. M. Standish*, JPL/CalTech
    !  * see http://ssd.jpl.nasa.gov/txt/aprx_pos_planets.pdf
    ! Standish's table 1 (in au and radians). Perturbations are zero.
      Data eph(1)%desc / "Keplerian Elements Valid 1800AD-2050AD." /
      Data eph(1)%n / 9 /
      Data eph(1)%lrad / .True. /
      Data eph(1)%epoch / 2451545.00D0 /
      Data eph(1)%jul1, eph(1)%jul2 / 2378497.0, 2470172.0 /
      Data (eph(1)%name(jcnt), j=1, 9) / "Mercury", "Venus", "Earth",&
     &  "Mars", "Jupiter", "Saturn", "Uranus", "Neptune", "Pluto" /
    ! This is Standish's table 1 (in au and radians). Perturbations are
    zero.
      Data ((eph(1)%o(icnt, jcnt), i=1, 16), j=1, 9) / &
     & 0.38709927, 0.20563594, 0.12225995, 4.4025989, 1.3518935, &
     & 0.84353095, 3.70000009E-07, 1.90600003E-05, - 1.03803286E-04, &
     & 2608.7903, 2.80085020E-03, - 2.18760967E-03, 0.0000000, &
     & 0.0000000, 0.0000000, 0.0000000, 0.72333568, 6.77671982E-03, &
     & 5.92482723E-02, 3.1761343, 2.2968962, 1.3383157, 3.90000014E-06, &
     & - 4.10700013E-05, - 1.37689030E-05, 1021.3286, 4.68322469E-05, - &
     & 4.84667765E-03, 0.0000000, 0.0000000, 0.0000000, 0.0000000, &
     & 1.0000026, 1.67112295E-02, - 2.67209913E-07, 1.7534375, &
     & 1.7966015, 0.0000000, 5.62000014E-06, - 4.39200012E-05, - &
     & 2.25962198E-04, 628.30756, 5.64218918E-03, 0.0000000, 0.0000000, &
     & 0.0000000, 0.0000000, 0.0000000, 1.5237104, 9.33941007E-02, &
     & 3.22832055E-02, - 7.94723779E-02, - 0.41789517, 0.86497712, &
     & 1.84700002E-05, 7.88199977E-05, - 1.41918135E-04, 334.06131, &
     & 7.75643345E-03, - 5.10636950E-03, 0.0000000, 0.0000000, &
     & 0.0000000, 0.0000000, 5.2028871, 4.83862385E-02, 2.27660220E-02, &
     & 0.60033119, 0.25706047, 1.7536005, - 1.16069998E-04, - &
```

```
      & 1.32529996E-04, - 3.20641411E-05, 52.966312, 3.70929041E-03, &
      & 3.57253314E-03, 0.0000000, 0.0000000, 0.0000000, 0.0000000, &
      & 9.5366764, 5.38617894E-02, 4.33887430E-02, 0.87186599, &
      & 1.6161553, 1.9837835, - 1.25059998E-03, - 5.09909994E-04, &
      & 3.37911442E-05, 21.336540, - 7.31244357E-03, - 5.03838016E-03, &
      & 0.0000000, 0.0000000, 0.0000000, 0.0000000, 19.189165, &
      & 4.72574383E-02, 1.34850740E-02, 5.4670362, 2.9837148, 1.2918390, &
      & - 1.96175999E-03, - 4.39700016E-05, - 4.24008576E-05, 7.4784222, &
      & 7.12186471E-03, 7.40122399E-04, 0.0000000, 0.0000000, 0.0000000, &
      & 0.0000000, 30.069923, 8.59048031E-03, 3.08930874E-02, - &
      & 0.96202600, 0.78478318, 2.3000686, 2.62910005E-04, &
      & 5.10499995E-05, 6.17357864E-06, 3.8128369, - 5.62719675E-03, - &
      & 8.87786155E-05, 0.0000000, 0.0000000, 0.0000000, 0.0000000, &
      & 39.482117, 0.24882729, 0.29914966, 4.1700983, 3.9107401, &
      & 1.9251670, - 3.15960002E-04, 5.17000008E-05, 8.40899645E-07, &
      & 2.5343544, - 7.09117157E-04, - 2.06556579E-04, 0.0000000, &
      & 0.0000000, 0.0000000, 0.0000000 /
!
! Approximate Positions of the Major Planets -
   ! Data and Approximation Model from E. M. Standish*, JPL/CalTech
   !  * see http://ssd.jpl.nasa.gov/txt/aprx_pos_planets.pdf
   ! Standish's table 2 (in au and radians). Perturbations are not zero.
     Data eph(2)%desc / "Keplerian Elements Valid 3000 BC - 3000 AD." /
     Data eph(2)%n / 9 /
     Data eph(2)%lrad / .True. /
     Data eph(2)%epoch / 2451545.00D0 /
     Data eph(2)%jul1, eph(2)%jul2 / 625674, 2816788 /
     Data (eph(2)%name(jcnt), j=1, 9) / "Mercury", "Venus", "Earth",&
    &  "Mars", "Jupiter", "Saturn", "Uranus", "Neptune", "Pluto" /
     Data ((eph(2)%o(icnt, jcnt), i=1, 16), j=1, 9) / &
    & 0.38709843, 0.20563661, 0.12227069, 4.4026222, 1.3518922, &
    & 0.84368551, 0.0000000, 2.12300001E-05, - 1.03002007E-04, &
    & 2608.7903, 2.78205727E-03, - 2.13177688E-03, 0.0000000, &
    & 0.0000000, 0.0000000, 0.0000000, 0.72332102, 6.76399004E-03, &
    & 5.93023673E-02, 3.1761451, 2.2997777, 1.3381896, - &
    & 2.60000007E-07, - 5.10700011E-05, 7.59113527E-06, 1021.3286, &
    & 9.91285546E-04, - 4.76024114E-03, 0.0000000, 0.0000000, &
    & 0.0000000, 0.0000000, 1.0000002, 1.67316291E-02, - &
    & 9.48516663E-06, 1.7534785, 1.7964685, - 8.92317668E-02, - &
    & 2.99999989E-08, - 3.66099994E-05, - 2.33381579E-04, 628.30762, &
    & 5.54932002E-03, - 4.21040738E-03, 0.0000000, 0.0000000, &
    & 0.0000000, 0.0000000, 1.5237124, 9.33651105E-02, 3.23203318E-02, &
    & - 7.97289312E-02, - 0.41743821, 0.86765921, 9.69999974E-07, &
    & 9.14900011E-05, - 1.26493964E-04, 334.06125, 7.89301097E-03, - &
    & 4.68663359E-03, 0.0000000, 0.0000000, 0.0000000, 0.0000000, &
    & 5.2024803, 4.85358983E-02, 2.26650927E-02, 0.59925520, &
    & 0.24914493, 1.7504400, - 2.86400009E-05, 1.80260002E-04, - &
    & 5.63216017E-05, 52.969063, 3.17635899E-03, 2.27322499E-03, - &
    & 2.17328397E-06, 1.05837814E-03, - 6.21955749E-03, 0.66935557, &
    & 9.5414991, 5.55082485E-02, 4.35327180E-02, 0.87398607, &
    & 1.6207365, 1.9833919, - 3.06500006E-05, - 3.20440013E-04, &
    & 7.88834659E-05, 21.329931, 9.45610274E-03, - 4.36594151E-03, &
    & 4.52022823E-06, - 2.34475732E-03, 1.52402408E-02, 0.66935557, &
    & 19.187979, 4.68574017E-02, 1.34910680E-02, 5.4838729, 3.0095420, &
    & 1.2908891, - 2.04550000E-04, - 1.54999998E-05, - 3.14429781E-05, &
    & 7.4786506, 1.61739404E-03, 1.00176642E-03, 1.01806800E-05, - &
    & 1.70574244E-02, 3.08735552E-03, 0.13387112, 30.069527, &
    & 8.95438995E-03, 3.08932904E-02, 5.3096914, 0.81474739, &
    & 2.3001058, 6.44699976E-05, 8.17999990E-06, 3.90953755E-06, &
    & 3.8129361, 1.76267436E-04, - 1.05819658E-04, - 7.21658762E-06, &
    & 1.19286822E-02, - 1.77369907E-03, 0.13387112, 39.486862, &
    & 0.24885239, 0.29916763, 4.1707320, 3.9112310, 1.9251275, &
    & 4.49750992E-03, 6.01600004E-05, 8.74410020E-08, 2.5338767, - &
    & 1.69092222E-04, - 1.41368364E-04, - 2.20386923E-04, 0.0000000, &
    & 0.0000000, 0.0000000 /
!
Contains
!
! Elements Routines
```

```
! requires  constants : s_ZERO s_ONE s_TWO s_D2PI s_DPC
!
      Subroutine Title
         Write (*,*) "Approximate Positions of the Major Planets"
         Write (*,*) "Method and Data from E. M. Standish, JPL/CalTech"
         Write (*,*) eph(2)%desc
         Write (*,*) "(http://ssd.jpl.nasa.gov/txt/aprx_pos_planets.pdf)"
         Write (*,*) SMODVER
         Write (*,*)
      End Subroutine
!
      Subroutine helio (np, jd, p, itbl)
    ! for planet np and julian date jd and using using table itbl,
    ! return j2000 ecliptic position (au) and velocity (au/yr).
    ! in cartesian coordinates (p(1)-p(6)).
         Implicit None
         Integer, Intent (In) :: np ! planet 1-9
         Real (8), Intent (In) :: jd ! julian date
         Real (8), Intent (Out) :: p (6)! position (au)/velocity (au/yr)
         Integer, Intent (Out) :: itbl !table used or error if zero
         Real (8) :: z (8)! elements a e i l w o ma ea
         Real (8) :: po (8)
         z = s_ZERO
         po = s_ZERO
         itbl = tbl (jd)
         If (itbl .Gt. 0) Then
            Call calcelements (np, jd, itbl, z)
            Call el2op (z, po)
            Call op2ec (z, po, p)
         End If
      End Subroutine
!
      Real (8) Function kepler (ma, ec)! solve kepler's equation ma = ea + ec*sin(ea)
         Implicit None ! acceptable accuracy for this calculation
         Real (8), Intent (In) :: ma, ec ! mean anomaly (ma) and eccentricity in rad
         Real (8) :: r, ea, tol ! max error in eccentric anomaly ea in rad
         Integer :: i, maxit ! max iterations (1-4 typical for ec<0.3)
         tol = 1.0d-08
         maxit = 12
         ea = ma + ec * Sin (ma)! starting value
         Do i = 1, maxit ! newton(-raphson) iterations
            r = (ma-ea+ec*Sin(ea)) / (s_ONE-ec*Cos(ea))
            ea = ea + r
            If (Abs(r) .Le. tol) Exit
         End Do
         kepler = modulo (ea, s_D2PI)! eccentric anomaly adjusted 0-2pi
      End Function
!
      Integer Function tbl (jd)
         Implicit None
    ! jd = julian date  (eg 2451545.0)
    ! itbl=1 jd in range of table 1 (1800ad-2050ad) - highest accuracy
    ! itbl=2 jd outside range of table 1 but in range of table 2 (3000bc-3000ad)
    ! itbl=0 3000bc<jd or jd>3000ad  julian date out of range for ephemeris.
         Real (8), Intent (In) :: jd ! julian
         tbl = 0
         If ((jd .Gt. eph(2)%jul1) .And. (jd .Lt. eph(2)%jul1)) tbl = 2
         If ((jd .Gt. eph(1)%jul1) .And. (jd .Lt. eph(1)%jul2)) tbl = 1
      End Function

      Subroutine calcelements (np, jd, itbl, z)
         Implicit None
    ! calculate current elements z(jd) for planet j from jpl data
    ! z(1) = a ; z(2) = e ; z(3) = i
    ! z(4) = l ; z(5) = w ; z(6) = o
    ! z(7) = ma ; z(8) = ea
         Integer, Intent (In) :: np, itbl ! planet , table
         Real (8), Intent (In) :: jd ! julian
         Real (8), Intent (Out) :: z (8)! elements for jd
```

```
          Integer :: i
          Real (8) :: t, tz
          t = (jd-eph(itbl)%epoch) / s_DPC ! centuries since epoch
          Do i = 1, 6             !a,e,i,l,w,o
             z (i) = eph(itbl)%o(i, np) + eph(itbl)%o(i+6, np) * t
          !   if (i>2) z(i) = modulo(z(i), s_d2pi)    !optional scaling
          End Do
          !perturbation term tz, nonzero for planets 5-9 if table 2 used
          tz = eph(itbl)%o(13, np) * t ** 2 + eph(itbl)%o(14, np) * Cos &
         & (eph(itbl)%o(16, np)*t) + eph(itbl)%o(15, np) * Sin &
         & (eph(itbl)%o(16, np)*t)
          z (7) = modulo ((z(4)-z(5)+tz), s_D2PI)! mean anomaly in z(7)
          z (8) = kepler (z(7), z(2))! eccentric anomaly in z(8)
        End Subroutine
!
! Coordinates Subroutines
! requires constants : s_zero  s_d2pi  s_sobl  s_cobl
!
        Subroutine el2op (z, po)
     !heliocentric coordinates for orbital plane from elements
          Implicit None
          Real (8), Intent (In) :: z (8)! elements a,e,i,l,w,o,ma,ea
          Real (8), Intent (Out) :: po (6)! coordinates and velocities
          Real (8) :: v, xp, yp, vx, vy, s1, c1, s2
     ! heliocentric orbital plane
          po = 0.0d0
          s1 = Sin (z(8))
          c1 = Cos (z(8))
          s2 = Sqrt (1.0d0-z(2)*z(2))
          v = s_D2PI / (Sqrt(z(1))*(1.0d0-z(2)*c1))! velocity au/yr
          po (1) = z (1) * (c1-z(2))! xp (plane of orbit)
          po (2) = z (1) * s1 * s2 ! yp
          po (4) = - v * s1 ! vxp
          po (5) = v * c1 * s2 ! vyp
        End Subroutine
!
        Subroutine op2ec (z, po, pe)
     !heliocentric coordinates j2000 ecliptic plane from orbital plane
          Implicit None
          Real (8), Intent (In) :: z (8)! elements a,e,i,l,w,o,ma,ea
          Real (8), Intent (In) :: po (6)! orbital plane coordinates
          Real (8), Intent (Out) :: pe (6)! j2000 ecliptic plane coordinates
          Real (8) :: s1, s2, s3, c1, c2, c3
     ! heliocentric au, au/yr
          s1 = Sin (z(5)-z(6))
          s2 = Sin (z(3))
          s3 = Sin (z(6))
          c1 = Cos (z(5)-z(6))
          c2 = Cos (z(3))
          c3 = Cos (z(6))
          pe (1) = (c1*c3-s1*s3*c2) * po (1) - (s1*c3+c1*s3*c2) * po (2)! xec
          pe (2) = (c1*s3+s1*c3*c2) * po (1) - (s1*s3-c1*c3*c2) * po (2)! yec
          pe (3) = s1 * s2 * po (1) + c1 * s2 * po (2)! zec
          pe (4) = (c1*c3-s1*s3*c2) * po (4) - (s1*c3+c1*s3*c2) * po (5)! vxec
          pe (5) = (c1*s3+s1*c3*c2) * po (4) - (s1*s3-c1*c3*c2) * po (5)! vyec
          pe (6) = s1 * s2 * po (4) + c1 * s2 * po (5)! vzec
        End Subroutine
!
        Subroutine ec2eq (pe, pq)
     ! converts cartesian heliocentric j2000 ecliptic to equatorial
          Implicit None
          Real (8), Intent (In) :: pe (6)!ecliptic
          Real (8), Intent (Out) :: pq (6)!equatorial
     ! requires constants s_sobl s_cobl (sin and cos of obliquity 23.43928 deg)
          pq (1) = pe (1)! xeq same as xec
          pq (2) = s_COBL * pe (2) - s_SOBL * pe (3)! yeq
          pq (3) = s_SOBL * pe (2) + s_COBL * pe (3)! zeq
          pq (4) = pe (4)! vxeq same as vxec
          pq (5) = s_COBL * pe (5) - s_SOBL * pe (6)! vyeq
```

```
         pq (6) = s_SOBL * pe (5) + s_COBL * pe (6)! vzeq
      End Subroutine
!
      Subroutine eq2ec (pq, pe)
    ! converts cartesian heliocentric equatorial to ecliptic
    ! requires constants s_sobl s_cobl (sin and cos of obliquity 23.43928 deg)
         Implicit None
         Real (8), Intent (Out) :: pe (6)  !ecliptic
         Real (8), Intent (In) :: pq (6)   !equatorial
         pe (1) = pq (1)! xec same as xeq
         pe (2) = s_COBL * pq (2) + s_SOBL * pq (3)! yec
         pe (3) = - s_SOBL * pq (2) + s_COBL * pq (3)! zec
         pe (4) = pq (4)! vxec same as vxeq
         pe (5) = s_COBL * pq (5) + s_SOBL * pq (6)! vyec
         pe (6) = - s_SOBL * pq (5) + s_COBL * pq (6)! vzec
      End Subroutine
!
      Subroutine sphere (x, y, z, rho, theta, phi)
    ! cartesian to spherical coordinates (angles in radians)
    ! distance (rho), longitude (theta), and latitude (phi)
    ! x = r cos(phi) cos (theta)  y = r cos(phi) sin(theta)  z = r sin(phi)
         Implicit None
         Real (8), Intent (In) :: x, y, z
         Real (8), Intent (Out) :: rho, theta, phi
         Real (8) :: r
         theta = s_ZERO
         phi = s_ZERO
         rho = Sqrt (x*x+y*y+z*z)
         r = Sqrt (x*x+y*y)
         If (r /= s_ZERO) Then
            theta = modulo (Atan2(y, x), s_D2PI)
            phi = Atan2 (z, r)
         End If
      End Subroutine
! ++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
! Copyright 2018 Cumulo Epsilon (epsilon0167) (GPG Key ID 8F126A52)

! Redistribution and use in source and binary forms, with or without
! modification, are permitted provided that the following conditions are met:

! 1. Redistributions of source code must retain the above copyright
! notice, this list of conditions and the following disclaimer.

! 2. Redistributions in binary form must reproduce the above copyright
! notice, this list of conditions and the following disclaimer in the
! documentation and/or other materials provided with the distribution.

! THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS
 ! "AS IS" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT
 ! LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS
 ! FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE
 ! COPYRIGHT HOLDER OR CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT,
 ! INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING,
 ! BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES;
 ! LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER
 ! CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT
 ! LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING
 ! IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF
 ! THE POSSIBILITY OF SUCH DAMAGE.
! ++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
End Module

```