REPORT DOCUMENTATION PAGE

Form Approved
OMB No. 0704-0188

1. REPORT DATE *(DD-MM-YYYY)*
15-06-2001

2. REPORT TYPE
UU

3. DATES COVERED *(From - To)*

4. TITLE AND SUBTITLE
See the abstract below

5a. CONTRACT NUMBER

5b. GRANT NUMBER

5c. PROGRAM ELEMENT NUMBER

6. AUTHOR(S)
See the abstract below

5d. PROJECT NUMBER

5e. TASK NUMBER

5f. WORK UNIT NUMBER

7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES)

8. PERFORMING ORGANIZATION REPORT NUMBER

9. SPONSORING / MONITORING AGENCY NAME(S) AND ADDRESS(ES)

10. SPONSOR/MONITOR'S ACRONYM(S)

11. SPONSOR/MONITOR'S REPORT NUMBER(S)

12. DISTRIBUTION / AVAILABILITY STATEMENT

DISTRIBUTION STATEMENT A
Approved for Public Release
Distribution Unlimited

13. SUPPLEMENTARY NOTES

14. ABSTRACT

Phone.txt DTICPUB0615.txt

US NAVAL OBSERVATORY COLLECTION

20010705 160

15. SUBJECT TERMS

16. SECURITY CLASSIFICATION OF:
UU

a. REPORT	b. ABSTRACT	c. THIS PAGE	17. LIMITATION OF ABSTRACT	18. NUMBER OF PAGES	19a. NAME OF RESPONSIBLE PERSON
					See the abstract above

19b. TELEPHONE NUMBER *(include area code) See the phone list above*

Beish, J. D. (1999). "Windows International Mars Patrol Astronomical Calculator WIMP," The Digital Lens: Newsletter of the Association of Lunar and Planetary Observers Computing Section, http://www.m2c3.com/alpocs/tdl1999/wimpinfo10031999/WIMP.html

Beish, J. D., (2000). "Discrete Topographic and Orographic Clouds of Mars," Association of Lunar and Planetary Observers Internet Web Page: The Mars Section, http://www.lpl.arizona.edu/~rhill/alpo/marstuff/discrete.htm

Beish, J. D. (2000). "A German Equatorial Mount for the Planetary Telescope," The Digital Lens: Newsletter of the Association of Lunar and Planetary Observers Computing Section, http://www.m2c3.com/alpocs/tdl2000/mountmath05222000/mount.htm

Beish, J. D. (2000). "Martian Volcanoes on HST Images," Association of Lunar and Planetary Observers Internet Web Page: The Mars Section, http://www.lpl.arizona.edu/~rhill/alpo/marstuff/VOLCANO.htm

Beish, J. D. (2000). "Measuring Celestial Dimensions with Micrometers," The Digital Lens: Newsletter of the Association of Lunar and Planetary Observers Computing Section, http://www.m2c3.com/alpocs/tdl2000/celestialdimensions01232000/ celestial_dimensions.htm

Beish, J. D. (2000). "Nuts and Bolts of Computing the Ephemeris - Part Four," The Digital Lens: Newsletter of the Association of Lunar and Planetary Observers Computing Section, http://www.m2c3.com/alpocs/tdl1999/nutsbolts08301999/TDL-4.html

Beish, J.D. (2000). "Practical Calculations for the Newtonian Secondary Mirror," The Digital Lens: Newsletter of the Association of Lunar and Planetary Observers Computing Section, http://www.m2c3.com/alpocs/tdl2000/telescopemath06202000/field1.htm

Beish, J. D. (2000). "Some of the 'Astronomical Seeing' Scales," The Digital Lens: Newsletter of the Association of Lunar and Planetary

Nuts and Bolts of Computing the Ephemeris Part Four

Jeff Beish

Association of Lunar and Planetary Observers (A.L.P.O.)

INTRODUCTION

In this installment we take a peek at the main program from beginning to end. Parts 13 dealt with some routines for time and date, so now it's time for the main course.

The following equations and their use can be found in the book Astronomical Algorithms, By Jean Meeus (Willmann-Bell, Inc. – ISBN 0-943396-35-2). Detailed discussion on these equations will not be given. This article will illustrate programming technique.

1: let's dimension statements to make room in memory for program variable array storage:

```
Dim DT

Dim Leap As Integer

Dim GoFlag As Integer

Dim UnivMonth As Variant

Dim UnivDay As Variant

Dim UnivYear As Variant

Dim txtUnivDate As String

Dim PASS As Integer, PASSf As Integer

Dim YearChange

Dim PRT As Integer, FRT As Integer
```

```
Dim PrPage
```

2: several short sub-routines discussed in ***Nuts and Bolts of Computing the Ephemeris - Part Three***

```
Function Arccos(X)

X = (Atn(-X / Sqr(-X * X + 1)) + 2 * Atn(1)) / rad

End Function

Function Arcsin(X)

X = Atn(X / Sqr(-X * X + 1)) / rad

End Function

Private Sub NORM(X)

X = ((X / 360) - Int(X / 360)) * 360

If X = 360 Then X = 0

End Sub
```

3: now let's tour the main body of our program.

Let's print out the header first (this can be used at the top of each page if desired):

```
        Printer.Print Tab(4); "Date"; Tab(12); "R.A."; Tab(18);
        "Dec"; Tab(23); "Dist"; Tab(30); "Ls"; Tab(36); "De"; Tab
        (43); "Ds"; Tab(47); "Phase"; Tab(53); "Defect"; Tab(60);
        "Axis";

        Tab(65); "Size"; Tab(71); "Mag"; Tab(76); "CM"

        Printer.Print Tab(2); "dd-mm-yy"; Tab(11); "hh:mm"; Tab(19);
        "o"; Tab(23); "A.U."; Tab(31); "o"; Tab(37); "o"; Tab(43);
        "o";

        Tab(49); "k"; Tab(55); "o"; Tab(61); "o"; Tab(67); ",,";

        Tab(71); "(v)"; Tab(77); "o"

Printer.Print Tab(2); String(75,"_")
```

4: Begin by **computing** the geocentric mean longitude of the Sun.

```
L = 279.696678 + 36000.76892 * T + 0.0003025 * T2

AH = 153.23 + 22518.7541 * T

BH = 216.57 + 45037.5082 * T

CH = 312.69 + 32964.3577 * T

DH = 350.74 + 445267.1142 * T - 0.00144 * T2

EH = 231.19 + 20.2 * T

HH = 353.4 + 65928.7155 * T

L = L + 0.00134 * Cos(rad * AH) + 0.00154 * Cos(rad * BH)

+ 0.002 * Cos(rad * CH) + 0.00179 * Sin(rad * DH) + 0.00178 * Sin(rad
* EH)

Call NORM(L)
```

5: We begin with **computing** the mean anomalies for Earth, Mars, Jupiter, a correction for Mars, and Venus:

```
MEarth = 358.475845 + 35999.04975 * T - 0.00015 * T2 - 0.0000033 * T3

Call NORM(MEarth)

mm = 319.51913 + 19139.85475 * T + 0.000181 * T2

Call NORM(mm)

MJ = 225.32833 + 3034.69202 * T - 0.0007220001 * T2

Call NORM(MJ)

DM = rad * (3 * MJ - 8 * mm + 4 * MEarth)

mm = mm - 0.01133 * Sin(DM) - 0.00933 * Cos(DM)

L1 = 293.737333 + 19141.69551 * T + 0.0003107 * T2

Call NORM(L1)

MV = 212.60322 + 58517.80387 * T + 0.001286 * T2

Call NORM(MV)
```

6: perturbations

```
L1 = L1 - 0.01133 * Sin(DM) - 0.00933 * Cos(DM)
+ 0.00705 * Cos(rad * (MJ - mm - 48.958))
+ 0.00607 * Cos(rad * (2 * MJ - mm - 188.35))
+ 0.00445 * Cos(rad * (2 * MJ - 2 * mm - 191.897))
+ 0.00388 * Cos(rad * (MEarth - 2 * mm + 20.495))
+ 0.00238 * Cos(rad * (MEarth - mm + 35.097))
+ 0.00204 * Cos(rad * (2 * MEarth - 3 * mm + 158.638))
+ 0.00177 * Cos(rad * (3 * mm - MV - 57.602))
+ 0.00136 * Cos(rad * (2 * MEarth - 4 * mm + 154.093))
+ 0.00104 * Cos(rad * (MJ + 17.618))
Call NORM(L1)
```

7: Calculate the distance to Mars from Sun:

```
A = 1.5236833 + 0.000053227 * Cos(rad * (MJ - mm + 41.1306))
+ 0.000050989 * Cos(rad * (2 * MJ - 2 * mm - 101.9847))
+ 0.000038278 * Cos(rad * (2 * MJ - mm - 98.3292))
+ 0.000015996 * Cos(rad * (MEarth - mm - 55.555))
+ 0.000014764 * Cos(rad * (2 * MEarth - 3 * mm + 68.622))
+ 0.000008966 * Cos(rad * (MJ - 2 * mm + 43.615))
+ 0.000007914 * Cos(rad * (3 * MJ - 2 * mm - 139.737))
+ 0.000007004 * Cos(rad * (2 * MJ - 3 * mm - 102.888))
+ 0.00000662 * Cos(rad * (MEarth - 2 * mm + 113.202))
+ 0.00000493 * Cos(rad * (3 * MJ - 3 * mm - 76.243))
+ 0.000004693 * Cos(rad * (3 * MEarth - 5 * mm + 190.603))
```

```
+ 0.000004571 * Cos(rad * (2 * MEarth - 4 * mm + 244.702))

+ 0.000004409 * Cos(rad * (3 * MJ - mm - 115.828))
```

8: Instead of using Kepler's iterations to compute the center of the Earth's orbit we will use the Equation of Center:

```
C = (1.91946 - 0.004789 * T - 0.000014 * T2) * Sin(rad * MEarth)

+ (0.020094 - 0.0001 * T) * Sin(rad * 2 * MEarth)

+ 0.000293 * Sin(rad * 3 * MEarth)

E1 = 0.09331289 + 0.000092064 * T - 0.000000077 * T2

E = mm + ((E1 / rad) * Sin(rad * mm)) / (1 - E1 * Cos(rad * mm))

NV = (2 / rad) * Atn(Tan(rad * E / 2) * Sqr((1 + E1) / (1 - E1)))

TH = L + C

Call NORM(TH)

NU = MEarth + C

Call NORM(NU)

E0 = 0.01675104 - 0.0000418 * T - 0.000000126 * T2
```

9: Radius vector of Sun.

```
RE = 1.00000023 * (1 - E0 * E0) / (1 + E0 * Cos(rad * NU))

+ 0.00000543 * Sin(rad * AH) + 0.00001575 * Sin(rad * BH)

+ 0.00001627 * Sin(rad * CH) + 0.00003076 * Cos(rad * DH)

+ 0.00000927 * Sin(rad * HH)

R = A * (1 - E1 * Cos(rad * E))

O1 = 48.786442 + 0.77099177 * T - 0.0000014 * T2 - 0.00000533 * T3

U = L1 + NV - mm - O1

Call NORM(U)

I1 = 1.850333 - 0.000675 * T + 0.0000126 * T2
```

```
SB = Sin(rad * U) * Sin(rad * I1)

Call Arcsin(SB): B = SB

LO = Atn(Cos(rad * I1) * Tan(rad * U)) / rad

If LO < 0 Then LO = LO + 360

If Abs(LO - U) > 45 Then LO = LO + 180

If LO > 360 Then LO = LO - 360

LM = LO + O1

Call NORM(LM)

DA = Sqr(RE * RE + R * R

+ (2 * R * RE) * Cos(rad * B) * Cos(rad * (LM - TH)))

LTim = (DA * 499.012) / 60

NO = R * Cos(rad * B) * Sin(rad * (LM - TH))

d = R * Cos(rad * B) * Cos(rad * (LM - TH)) + RE

LT = Atn(NO / d) / rad

If d < 0 Then LT = LT + 180

LA = LT + TH

Call NORM(LA)

BA = R * Sin(rad * B) / DA

Call Arcsin(BA)

EM = Cos(rad * BA) * Cos(rad * LT)

Call Arccos(EL)

EP = 23.452294 - 0.0130125 * T - 0.00000164 * T2 + 0.000000503 * T3

CE = Cos(rad * EP)

SE = Sin(rad * EP)

RX = Sin(rad * LA) * CE - Tan(rad * BA) * SE
```

```
RY = Cos(rad * LA)

RA = Atn(RX / RY) / rad

If RY < 0 Then RA = RA + 180

Call NORM(RA)

RH = Int(RA / 15)

RM = 4 * (RA - 15 * RH)

RS = (RM - Int(RM)) * 60

RM = Int(RM)

RS = Int(RS)

RightAsc = Format(TimeSerial(RH, RM, RS), "hh:nn")

DC = Sin(rad * BA) * CE + Cos(rad * BA) * SE * Sin(rad * LA)

Call Arcsin(DC)

If DC < 0 Then DCGN = "-" Else DCGN = " "

N1 = Sin(rad * I1) * Sin(rad * O1)

N2 = Cos(rad * I1) * SE + Sin(rad * I1) * CE * Cos(rad * O1)

N0 = Atn(N1 / N2) / rad

If N2 < 0 Then N0 = N0 + 180

J1 = Cos(rad * I1) * CE - Sin(rad * I1) * SE * Cos(rad * O1)

Call Arccos(J1): j = J1

O4 = SE * Sin(rad * O1)

O5 = Sin(rad * I1) * CE + Cos(rad * I1) * SE * Cos(rad * O1)

O6 = Atn(O4 / O5) / rad

If O5 < 0 Then O6 = O6 + 180

Call NORM(O6)

TT = UnivYear - 1905
```

```
A0 = 316.55 + 0.006751 * TT

D0 = 52.85 + 0.00348 * TT

O7 = Cos(rad * D0) * Cos(rad * (N0 - A0))

O8 = -Sin(rad * D0) * Sin(rad * j) + Cos(rad * D0) * Cos(rad * j)

* Sin(rad * (N0 - A0))

O9 = Atn(O7 / O8) / rad

If O8 < 0 Then O9 = O9 + 180

O9 = O9 - O6

D4 = Sin(rad * j) * Cos(rad * (N0 - A0))

D5 = Cos(rad * D0) * Cos(rad * j) - Sin(rad * D0) * Sin(rad * j) *
Sin(rad * (N0 - A0))

D6 = Atn(D4 / D5) / rad

If D5 < 0 Then D6 = D6 + 180

Ls = LM - O1 - O9

Call NORM(Ls)

I6 = Sin(rad * D0) * Cos(rad * j) + Cos(rad * D0) * Sin(rad * j) *
Sin(rad * (N0 - A0))

Call Arccos(I6): I7 = I6

A6 = rad * (A0 - RA)

De = -Sin(rad * D0) * Sin(rad * DC) - Cos(rad * D0) * Cos(rad * DC)

* Cos(A6)

Call Arcsin(De)

D7 = Cos(rad * De)

DI = 9.359999 / DA

P6 = Cos(rad * D0) * Sin(A6)

P7 = Sin(rad * D0) * Cos(rad * DC) - Cos(rad * D0) * Sin(rad * DC)
```

```
* Cos(A6)
P8 = Atn(P6 / P7) / rad
If P7 < 0 Then P8 = P8 + 180
Call NORM(P8)
A1 = (Cos(rad * D0) * Sin(rad * DC) - Sin(rad * D0) * Cos(rad * DC)
* Cos(A6)) / D7
A2 = -Cos(rad * DC) * Sin(A6) / D7
A3 = Atn(A1 / A2) / rad
If A2 < 0 Then A3 = A3 + 180
A4 = A3 - D6
Call NORM(A4)
D1 = Sin(rad * Ls) * Sin(rad * I7)
D2 = Atn(D1 / Sqr(-D1 * D1 + 1)) / rad
If Ds < 0 Then DSGN = "-" Else DSGN = " "
D3 = Cos(rad * D2)
A6 = Sin(rad * Ls) * Cos(rad * I7) / D3
A7 = Cos(rad * Ls) / D3
A8 = Atn(A6 / A7) / rad
If A7 < 0 Then A8 = A8 + 180
Call NORM(A8)
CI = (R * R + DA * DA - RE * RE) / (2 * R * DA)
If CI > 1 Then CI = 1
PD = 0.5 * (1 - CI): PE = D1 * PD
If CI < 1 Then
I2 = (-Atn(CI / Sqr(-CI * CI + 1)) + 1.570796327) / rad
```

```
Else
I2 = 0
End If
Phase = 1 - PD
MCM = 350.891962 / 86400!
V1 = 350.891962 * (JulianDate - 2418322)
Call NORM(V1)
V1 = 329.499 + V1: If V1 > 360 Then V1 = V1 - 360
CM = V1 - A4 - 180 - 2.026612 * DA + (DT * MCM)
Call NORM(CM)
MG = -1.52 + 2.171472 * Log(R * DA) + 0.01486 * I2
If MG < 0 Then MGGN = "-" Else MGGN = " "
S2 = Atn(Sin(rad * TH) * CE / Cos(rad * TH)) / rad
If S2 < 0 Then S2 = S2 + 360
If Abs(S2 - TH) > 45 Then S2 = S2 + 180
If S2 > 360 Then S2 = S2 - 360
S3 = SE * Sin(rad * TH)
Call Arcsin(S3)
A9 = rad * (S2 - RA)
SD = Sin(rad * EL)
T6 = Cos(rad * S3) * Sin(A9) / SD
T7 = (Sin(rad * S3) * Cos(rad * DC) - Cos(rad * S3) * Sin(rad * DC)
* Cos(A9)) / SD
T8 = Atn(T6 / T7) / rad
Call NormAtn(T7, T8)
```

```
Q = T8 + 180

Call NORM(Q)
```

10: Now, print each line of the daily listing.

```
Printer.Print Tab(2); txtUnivDate;

Tab(11); RightAsc;

Tab(17); DCGN + Format(Abs(DC), "00.0");

Tab(23); Format(DA, "0.000");

Tab(29); Format(Ls, "000.0");

Tab(35); DEGN + Format(Abs(De), "00.0");

Tab(42); Format(Abs(D2), "00.0");

Tab(47); Format(Phase, "0.000");

Tab(53); Format(Q, "000.0");

Tab(59); Format(P8, "000.0");

Tab(65); Format(DI, "00.0");

Tab(70); MGGN + Format(Abs(MG), "0.0");

Tab(75); Format(CM, "000.0")

End Sub
```

I encourage you to type in the source code and generate your own**ephemeris**.

Click Here to Go Back to The Digital Lens