

1. MoonPhase. These functions have been stolen from the Perl module `Astro::MoonPhase`. See that module's documentation for full authors and references. Code originally derived from `moontool.c` by John Walker. This transcription of the Perl sources into C was performed by Bret Whissel, 11 March 2013.

```
⟨Includes 2⟩
⟨Functions 6⟩
```

2. `⟨Includes 2⟩ ≡`
`#include <math.h>`
`#include <time.h>`

This code is used in section 1.

3. Astronomical constants.

```
#define Epoch ((double) 2444238.5) /* 1980 January 0.0 */
```

4. Constants defining the Sun's apparent orbit.

```
#define Elonge ((double) 278.833540) /* ecliptic longitude of the Sun at epoch 1980.0 */
#define Elongp ((double) 282.596403) /* ecliptic longitude of the Sun at perigee */
#define Eccent ((double) 0.016718) /* eccentricity of Earth's orbit */
```

5. Elements of the Moon's orbit, epoch 1980.0.

```
#define Mmlong ((double) 64.975464) /* moon's mean longitude at the epoch */
#define Mmlongp ((double) 349.383063) /* mean longitude of the perigee at the epoch */
#define Mlnode ((double) 151.950429) /* mean longitude of the node at the epoch */
#define Synmonth ((double) 29.53058868) /* synodic month (new Moon to new Moon) */
```

6. Handy mathematical definitions.

```
#define Pi ((double) 3.14159265358979323846) /* assume not near black hole nor in Tennessee */
#define torad(x) ((x) * (Pi/180.0))
#define todeg(x) ((x) * (180.0/Pi))
#define INV360 ((double) 1.0/(double) 360.0)
```

```
⟨Functions 6⟩ ≡
static double fixangle(double x)
{
    return x - 360.0 * floor(x * INV360);
}
```

See also sections 7, 8, and 9.

This code is used in section 1.

7. `jtime`: convert internal date and time to astronomical Julian time (i.e. Julian date plus day fraction). Calculates $(seconds / (seconds \text{ per day})) + \text{julian date of UNIX epoch}$.

```
⟨Functions 6⟩ +=
static double jtime(time_t *t)
{
    double julian;
    julian ← ((double) *t / (double) 86400.0) + 2440587.5;
    return julian;
}
```

8. *kepler*: solve the equation of Kepler.

```

⟨Functions 6⟩ +=
double kepler(double m, double ecc)
{
    double e, delta, epsilon ← 1 · 10-6;
    m ← torad(m);
    e ← m;
    do {
        delta ← e - ecc * sin(e) - m;
        e -= delta / (1 - ecc * cos(e));
    } while (fabs(delta) > epsilon);
    return e;
}

```

9. *moonphase*: Calculate phase of moon. The argument *thetime* is the UNIX time for which to calculate. Returns *pphase*, the illuminated fraction of the Moon's disc; *mage*, the Moon's age in days; *mpfrac*, the terminator phase angle as a percentage of a full circle (i.e., 0 to 1).

```

⟨Functions 6⟩ +=
void moonphase(time_t *thetime, double *pphase, double *mage, double *mpfrac)
{
    double pdate, Day, N, M, Ec, Lambdasun;
    double ml, MM, Ev, Ae, A3, MmP, mEc, A4, lP, V, lPP;
    double MoonAge, MoonPhase;

    pdate ← jtime(thetime);
    ⟨Calculation of the Sun's position 10⟩
    ⟨Calculation of the Moon's position 11⟩
    MoonAge ← lPP - Lambdasun; /* Age of the Moon in degrees. */
    MoonPhase ← (1.0 - cos(torad(MoonAge)))/2.0; /* Phase of the Moon. */
    *pphase ← MoonPhase; /* illuminated fraction */
    *mage ← Synmonth * (fixangle(MoonAge)/360.0); /* age of moon in days */
    *mpfrac ← fixangle(MoonAge)/360.0;
}

```

10. ⟨Calculation of the Sun's position 10⟩ ≡
 $Day \leftarrow pdate - Epoch;$ /* date within epoch */
 $N \leftarrow \text{fixangle}((360.0/365.2422) * Day);$ /* mean anomaly of the Sun */
 $M \leftarrow \text{fixangle}(N + Elong_e - Elong_p);$ /* convert from perigee co-ordinates to epoch 1980.0 */
 $Ec \leftarrow \text{kepler}(M, Eccent);$ /* solve equation of Kepler */
 $Ec \leftarrow \text{sqrt}((1.0 + Eccent)/(1.0 - Eccent)) * \tan(Ec/2.0);$
 $Ec \leftarrow 2.0 * \text{todeg}(\text{atan}(Ec));$ /* true anomaly */
 $Lambdasun \leftarrow \text{fixangle}(Ec + Elong_p);$ /* Sun's geocentric ecliptic longitude */

This code is used in section 9.

11. \langle Calculation of the Moon's position 11 $\rangle \equiv$

```

ml ← fixangle(13.1763966 * Day + Mmlong);    /* Moon's mean longitude. */
MM ← fixangle(ml - 0.1114041 * Day - Mmlongp); /* Moon's mean anomaly. */
Ev ← 1.2739 * sin(torad(2.0 * (ml - Lambdasun) - MM)); /* Evecton. */
Ae ← 0.1858 * sin(torad(M)); /* Annual equation. */
A3 ← 0.37 * sin(torad(M)); /* Correction term. */
MmP ← MM + Ev - Ae - A3; /* Corrected anomaly. */
mEc ← 6.2886 * sin(torad(MmP)); /* Correction for the equation of the centre. */
A4 ← 0.214 * sin(torad(2.0 * MmP)); /* Another correction term. */
lP ← ml + Ev + mEc - Ae + A4; /* Corrected longitude. */
V ← 0.6583 * sin(torad(2.0 * (lP - Lambdasun))); /* Variation. */
lPP ← lP + V; /* True longitude. */

```

This code is used in section 9.

12. Create a header file for external linking.

\langle moonphase.h 12 $\rangle \equiv$

```

void moonphase(time_t *, double *, double *, double *);

```

13. Create a small test program.

```

<mpctest.c 13> ≡
#include <stdio.h>
#include <stdlib.h>
#include <math.h>
#include <time.h>
#include "moonphase.h"

int main(int argc, char **argv)
{
    struct tm theday;
    time_t now ← time(0);
    double phase, age, frac;
    char *oldtz;
    int i;

    oldtz ← getenv("TZ");
    setenv("TZ", "", 1);
    tzset();
    theday.tm_year ← 2013 - 1900;
    theday.tm_mon ← 0;
    theday.tm_mday ← 1;
    theday.tm_hour ← 0;
    theday.tm_min ← 0;
    theday.tm_sec ← 0;
    now ← mktime(&theday);
    for (i ← 0; i ≤ 3 * 30 * 24; i++) { /* Generate 3 months (avg) of data, hourly */
        moonphase(&now, &phase, &age, &frac);
        localtime_r(&now, &theday);
        printf("%02d/%02d/%4d %02d: %5.3f; %6.3f days; %5.3f\n",
            theday.tm_mon + 1, theday.tm_mday, theday.tm_year + 1900, theday.tm_hour, phase, age, frac);
        now += 3600;
    }
    if (oldtz) setenv("TZ", oldtz, 1);
    else unsetenv("TZ");
    tzset();
}

```

14. Index.

Ae: [9](#), [11](#).
age: [13](#).
argc: [13](#).
argv: [13](#).
atan: [10](#).
A3: [9](#), [11](#).
A4: [9](#), [11](#).
cos: [8](#), [9](#).
Day: [9](#), [10](#), [11](#).
delta: [8](#).
e: [8](#).
Ec: [9](#), [10](#).
ecc: [8](#).
Eccent: [4](#), [10](#).
Elonge: [4](#), [10](#).
Elongp: [4](#), [10](#).
Epoch: [3](#), [10](#).
epsilon: [8](#).
Ev: [9](#), [11](#).
fabs: [8](#).
fixangle: [6](#), [9](#), [10](#), [11](#).
floor: [6](#).
frac: [13](#).
getenv: [13](#).
i: [13](#).
INV360: [6](#).
jtime: [7](#), [9](#).
julian: [7](#).
kepler: [8](#), [10](#).
Lambdasun: [9](#), [10](#), [11](#).
localtime_r: [13](#).
lP: [9](#), [11](#).
lPP: [9](#), [11](#).
M: [9](#).
m: [8](#).
mage: [9](#).
main: [13](#).
mEc: [9](#), [11](#).
mktime: [13](#).
ml: [9](#), [11](#).
Mlnode: [5](#).
MM: [9](#), [11](#).
Mmlong: [5](#), [11](#).
Mmlongp: [5](#), [11](#).
MmP: [9](#), [11](#).
MoonAge: [9](#).
MoonPhase: [9](#).
moonphase: [9](#), [12](#), [13](#).
mpfrac: [9](#).
N: [9](#).
now: [13](#).
oldtz: [13](#).
pdate: [9](#), [10](#).
phase: [13](#).
Pi: [6](#).
pphase: [9](#).
printf: [13](#).
setenv: [13](#).
sin: [8](#), [11](#).
sqrt: [10](#).
Synmonth: [5](#), [9](#).
t: [7](#).
tan: [10](#).
theday: [13](#).
thetime: [9](#).
time: [13](#).
tm: [13](#).
tm_hour: [13](#).
tm_mday: [13](#).
tm_min: [13](#).
tm_mon: [13](#).
tm_sec: [13](#).
tm_year: [13](#).
todeg: [6](#), [10](#).
torad: [6](#), [8](#), [9](#), [11](#).
tzset: [13](#).
unsetenv: [13](#).
V: [9](#).
x: [6](#).

⟨ Calculation of the Moon's position 11 ⟩ Used in section 9.
⟨ Calculation of the Sun's position 10 ⟩ Used in section 9.
⟨ Functions 6, 7, 8, 9 ⟩ Used in section 1.
⟨ Includes 2 ⟩ Used in section 1.
⟨ moonphase.h 12 ⟩
⟨ mptest.c 13 ⟩

MOONPHASE

	Section	Page
MoonPhase	1	1
Index	14	5