```
using namespace std;
#include <string>
#include <iostream>
class Airport
{
public:
	string airCode;
	void setAirCode(string a);
	string getAirCode();

	double latitude;
	void setLatitude(double lat);
	double getLatitude();

	double longitude;
	void setLongitude(double lon);
	double getLongitude();

	char timeZone;
	void setTimeZone(char a);
	char getTimeZone();

	Airport();
	Airport(string a, double b ,double c,char d);

	string toString();

};
Airport::Airport()
{
	airCode = "";
	latitude = 0;
	longitude = 0;
	timeZone = 'z';
}
Airport::Airport(string a, double b,double c,char d)
{
	airCode = a;
	latitude = b;
	longitude = c;
	timeZone = d;

}
void Airport::setAirCode(string a)
{
	airCode = a;
}
void Airport::setLatitude(double lat)
{
	latitude = lat;
}
void Airport::setLongitude(double lon)
{
	longitude = lon;
}
void Airport::setTimeZone(char a)
{
	timeZone = a;
```

```
}
string Airport::getAirCode()
{
	return airCode;
}
double Airport::getLatitude()
{
	return latitude;
}
double Airport::getLongitude()
{
	return longitude;
}
char Airport::getTimeZone()
{
	return timeZone;
}
string Airport::toString()
{
	string output = "";
	output += airCode + " " + to_string(latitude) + " " + to_string(longitude) + " " +
timeZone;
	return output;
}
```

```
// These functions include various date manipulation algorithms

//-----------------------------------------------------------------
// This function receives an integer year and returns true if the
// year is a leap year and false otherwise.
bool leapYear(int theYear);

//-----------------------------------------------------------------
// This function receives an integer month and year and returns an
// integer of the number of days in the month.  Leap years are
// considered.
int DaysInMonth(int theMonth, int theYear);

//-----------------------------------------------------------------
// This function receives a valid calendar date and returns the Julian
// date (the day number of the date in that year).
int julianDate(int theMonth, int theDay, int theYear);

//-----------------------------------------------------------------
// This method receives a calendar date and returns a boolean value
// defining the validity of the date.
bool validDate(int mon, int day, int yr);

//-----------------------------------------------------------------
// This function receives a valid date and returns a date code for the
// day of the week.  It counts the number of days since 1/1/1900
// which was on a Sunday.  Output is: 0=Sun,1=Mon, ..., 6=Sat.
int weekDay(int mon, int day, int year);

//-----------------------------------------------------------------
// This function receives a day code and returns the string (2nd
// parametere with the three-character day of the week descriptor
// (0=Sun,1=Mon, ..., 6=Sat)
void dayCode(int code, char descript[]);
```

```
// Sunrise/Sunset module
//
// Author:  T. Klingler
//----------------------------------------------------------------

// Misc constants (as needed)
//const double pi = 3.1415962536;

//----------------------------------------------------------------
// Time zone data type:
//    E Eastern time zone
//    C Central time zone
//    M Mountain time zone
//    P Pacific time zone
//    U Universal (Greenwich Mean) time

//----------------------------------------------------------------

// Texttime data type (time returned as string)
typedef char TextTime[8];

//----------------------------------------------------------------//
// This function receives a date and geographical coordinates   //
// returns the sunrise and sunset for that day and location     //
// Input:                                                       //
//        latitude:   Latitude of site (pos. float in degrees)  //
//        longitude:  Longitude of site (pos. float in degrees) //
//        mon:        Month (integer 1..12)                     //
//        day:        Day of month (1..31)                      //
//        year:       Year (4-digit integer 19xx or 20xx)       //
//        timeZone:   Code for time zone                        //
//        DST         bool indicating if daylight savings time  //
//                    is active.                                //
//    Output:                                                   //
//        sun_rise    Sunrise value (string)                    //
//        sun_set     Sunset value (string)                     //
//    Preconditions:                                            //
//        * Date is valid; year is four-digit                   //
//        * Latitude and longitude is 0.0 .. 90.0 degrees       //
//        * Time zone codes must be 'E', 'C', 'M', or 'P' only) //
//    Postconditions:                                           //
//        * Date and geog. coords. unchanged                    //
//                                                              //
//  Algorithm extracted from a BASIC program in Weatherwise     //
//  Magazine.  It was converted to Pascal in 1995 and then to   //
//  this C++ version.                                           //

void Sun_Rise_Set(  double latitude, double longitute,
                    int mon, int day, int year,
                    char timeZone, bool DST,
                    TextTime sun_rise, TextTime sun_set);
```

```cpp
// This file includes implementations for date functions
#define _CRT_SECURE_NO_WARNINGS
#include <iostream>
#include <string>
using namespace std;

#include "datefun.h"

//-----------------------------------------------------------------
// This function receives an integer year and returns true if the
// year is a leap year and false otherwise.

bool leapYear(int theYear)
{
  if ( theYear % 400 == 0 ||
     ( theYear % 4 == 0 && theYear % 100 != 0 ) )
     return true;
  else
     return false;

}  // end function leapYear

//-----------------------------------------------------------------
// This function receives an integer month and year and returns an
// integer of the number of days in the month.  Leap years are
// considered.

int DaysInMonth(int theMonth, int theYear)
{
  int days = 0;

  // 31 Day theMonths
  if (theMonth == 1 || theMonth == 3 || theMonth == 5  ||
      theMonth == 7 || theMonth == 8 || theMonth == 10 ||
      theMonth == 12 )
         days = 31;

  // 30 Day theMonths
  else if (theMonth == 4 || theMonth == 6 ||
           theMonth == 9 || theMonth == 11 )
        days = 30;

  // February
  else  // theMonth == 2
     if ( leapYear(theYear) )
        days = 29;
     else
        days = 28;

  return days;

}  // end function DaysInMonth

//-----------------------------------------------------------------
// This function receives a valid calendar date and returns the Julian
// date (the day number of the date in that year).

int julianDate(int theMonth, int theDay, int theYear)
```

```
{
  int dayCnt = 0;
  int mon;
  for (mon = 1; mon < theMonth; mon++)
     dayCnt += DaysInMonth(mon,theYear);
  dayCnt += theDay;

  return dayCnt;

}  // end function julianDate

//---------------------------------------------------------------------
// This function validates a calendar date and returns 'true' if
// all three components  represent a valid date and 'false' otherwise

bool validDate(int mon, int day, int yr)
{
   bool valDate = true;   // Assume a good date

   // Test for conditions that would make the date validity false
   if (yr < 1900)
      valDate = false;
   if ((mon < 1) || (mon > 12) || (day < 1) || (day > 31))
      valDate = false;
   else if (((mon == 4) || (mon == 6) || (mon == 9) || (mon == 11)) && (day == 31))
      valDate = false;
   else if ((mon == 2)  && leapYear(yr) && (day > 29))
     valDate = false;
   else if ((mon == 2)  && ! leapYear(yr) && (day > 28))
     valDate = false;

  return valDate;
}  // end function validDate

//---------------------------------------------------------------------
// This function receives a valid date and returns a date code for the
// day of the week.  It counts the number of days since 1/1/1900
// which was on a Sunday.  Output is: 0=Sun,1=Mon, ..., 6=Sat.

int weekDay(int mon, int day, int year)
{
    int DayCnt;
    int daynum,i;

    DayCnt = (year - 1900) * 365;
    DayCnt += ((year - 1900) / 4) + 1;
    for (i=1;i<=mon-1;i++)
    switch(i)
    {
        case 2:     DayCnt +=28; break;
        case 4:
        case 6:
        case 9:
        case 11:     DayCnt +=30; break;
        default:    DayCnt +=31;
    };

    if (((year - 1900) % 4 == 0) && (mon <= 2))
```

```
        DayCnt--;
    DayCnt += day;
    daynum = (DayCnt - 1) % 7;

    return daynum;
} // end function weekDay

//---------------------------------------------------------------------
// This function receives a day code and returns the string (2nd
// parametere with the three-character day of the week descriptor
// (0=Sun,1=Mon, ..., 6=Sat)
void dayCode(int code, char descript[])
{
     switch(code)
     {
          case 0: strcpy(descript, "SUN"); break;
          case 1: strcpy(descript, "MON"); break;
          case 2: strcpy(descript, "TUE"); break;
          case 3: strcpy(descript, "WED"); break;
          case 4: strcpy(descript, "THU"); break;
          case 5: strcpy(descript, "FRI"); break;
          case 6: strcpy(descript, "SAT"); break;
     }; // end switch
}
```

```
#include <iostream>
#include <iomanip>
#include <fstream>
#include <string>
#include "airports.h"
#include "datefun.h"
#include "sun.h"
/*
Created by Nathan Gaffney
CST 280 ADV C++
Program1
THis program displays the sunrise and
sunset of the week of a given date.
*/

using namespace std;
const int MAXARRAY = 100;
/*Function Prototypes*/
void buildLstAirports(Airport a[], int& numNumbers);
void validateData(int year, int month, int day, string airCode, Airport lstAirports[],
int numElems);
bool isDST(char day[], int month, int weekdayValue, int dayNumber );
int main()
{
	/************/
	//For the Continue Loop
	char answer;
	bool cont = true;
	/************/
	int numElems = 0;
	int enteredDate, yearInt, monthInt, dayInt;
	string airCode;
	bool DST= false;
	int dayNum;
	char day[3];
	double latitude, longitude;
	char timeZone;
	TextTime sunRise, sunSet;
	Airport lstAirports[MAXARRAY];
	buildLstAirports(lstAirports, numElems);

	do
	{
		cout << "Enter a date in the format of yyyymmdd: ";
		cin >> enteredDate;
		yearInt = enteredDate / 10000;
		monthInt = enteredDate%10000/100;
		dayInt = enteredDate % 100;

		cout << "Enter an airport three character code: ";
		cin >> airCode;
		for (int i = 0; i < 3; i++)
		{
			airCode[i] = toupper(airCode[i]);
		}
		validateData(yearInt, monthInt, dayInt, airCode, lstAirports, numElems);
		dayNum = weekDay(monthInt, dayInt, yearInt);
		DST = isDST(day, monthInt, dayNum, dayInt);
```

```
		for (int i = 0; i < numElems ; i++)
		{
			if (airCode == lstAirports[i].getAirCode())
			{
				latitude = lstAirports[i].getLatitude();
				longitude = lstAirports[i].getLongitude();
				timeZone = lstAirports[i].getTimeZone();
			}
		}


		/*****************Output*****************/
		for (int i = dayNum; i > 0; i--)
		{

			if (dayInt - 1 == 0)
			{
				if (monthInt - 1 == 0)
				{
					yearInt--;
					monthInt = 12;
					dayInt = DaysInMonth(monthInt, yearInt) - 1;
				}
				else
				{
					monthInt--;
					dayInt = DaysInMonth(monthInt, yearInt) - 1;
				}

			}
			else
			{
				dayInt--;
			}
		}
		for (int i = 0; i <= 6; i++)
		{
			if (dayInt > DaysInMonth(monthInt, yearInt))
			{
				if (monthInt  >= 12)
				{
					yearInt++;
					monthInt = 1;
				}
				else
				{
					monthInt++;
				}
				dayInt = 1;
			}
			dayCode(weekDay(monthInt, dayInt, yearInt), day);
			cout << day << " ";
			cout << setw(2) << setfill('0') << dayInt; //Creates a 0 infront of
single digits
			switch (monthInt)
			{
			case 1:
				cout << " JAN ";
```

```
                        break;
                case 2:
                        cout << " FEB ";
                        break;
                case 3:
                        cout << " MAR ";
                        break;
                case 4:
                        cout << " APR ";
                        break;
                case 5:
                        cout << " MAY ";
                        break;
                case 6:
                        cout << " JUN ";
                        break;
                case 7:
                        cout << " JUL ";
                        break;
                case 8:
                        cout << " AUG ";
                        break;
                case 9:
                        cout << " SEP ";
                        break;
                case 10:
                        cout << " OCT ";
                        break;
                case 11:
                        cout << " NOV ";
                        break;
                case 12:
                        cout << " DEC ";
                        break;
                default:
                        break;
                }
                Sun_Rise_Set(latitude, longitude, monthInt, dayInt, yearInt,
timeZone, DST, sunRise, sunSet);
                cout << yearInt << "  ";
                cout << "Rise:  " << sunRise << " am" << "  ";
                cout << "Set:  " << sunSet << " pm" << endl;
                dayInt++;
        }
        /***********Continuation Loop***************/
        cout << "Do you want to continue? Y/N: ";
        cin >> answer;
        if (answer != 'Y' && answer != 'y')
        {
                if (answer == 'N' || answer== 'n'){ cont = false; }

                else
                {
                        cout << "Incorrect character entered. Exiting." << endl;
                        cont = false;
                }
        }
```

```
        } while (cont);
        system("pause");
    return 0;
}
void buildLstAirports(Airport a[], int& numNumbers)
{
        int i = 0;
        string string;
        char char1;
        double number1, number2;

        ifstream fileIn;
        fileIn.open("cityinfo.txt");

        if (fileIn.fail())
        {
                cout << "Problem opening file";
                exit(-1);
        }

        fileIn >> string >> number1 >> number2 >> char1;
        a[i].setAirCode(string);
        a[i].setLatitude(number1);
        a[i].setLongitude(number2);
        a[i].setTimeZone(char1);


        while (!fileIn.eof() && i < MAXARRAY)    // Test for end of file and array
        {

                i++;
                fileIn >> string >> number1 >> number2 >> char1;
                a[i].setAirCode(string);
                a[i].setLatitude(number1);
                a[i].setLongitude(number2);
                a[i].setTimeZone(char1);
        }
        numNumbers = i;
}
void validateData(int year, int month, int day, string airCode, Airport list[], int
numElems)
{
        bool found = false;
        if (!validDate(month, day, year))
        {
                cout << "Invalid date! Exiting.";
                system("pause");
                exit(1);
        }
        for (int i = 0; i < numElems && !found; i++)
        {
                if (airCode == list[i].getAirCode()){ found = true; }
        }
        if (!found)
        {
                cout << "Invalid Airport Code! Exiting.";
                system("pause");
                exit(1);
```

```
	}

}
bool isDST(char day[], int month, int weekdayValue, int dayNumber)
{
	bool isDST = false;

	if (month == 4 || month == 5 || month == 6 || month == 7 || month == 8)
	{
		isDST = true;
	}
	else
	{
		if (month == 3)
		{
			switch (weekdayValue)
			{
			case 0:	if (dayNumber >= 8)
				isDST = true;

				break;
			case 1:	if (dayNumber >= 14)
				isDST = true;

				break;
			case 2:	if (dayNumber >= 13)
				isDST = true;

				break;
			case 3:	if (dayNumber >= 12)
				isDST = true;

				break;
			case 4:	if (dayNumber >= 11)
				isDST = true;

				break;
			case 5:	if (dayNumber >= 10)
				isDST = true;

				break;
			case 6:	if (dayNumber >= 9)
				isDST = true;

				break;
			};
		}
		else
		{
			switch (weekdayValue)
			{
			case 0:	if (dayNumber <= 1)
				isDST = false;
				break;
			case 1:	if (dayNumber <= 7)
				isDST = true;

				break;
```

```
                        case 2:   if (dayNumber <= 6)
                                  isDST = true;

                                  break;
                        case 3:   if (dayNumber <= 5)
                                  isDST = true;

                                  break;
                        case 4:   if (dayNumber <= 4)
                                  isDST = true;

                                  break;
                        case 5:   if (dayNumber <= 3)
                                  isDST = true;

                                  break;
                        case 6:   if (dayNumber <= 2)
                                  isDST = true;

                                  break;
                        };
                  }
            }
            return isDST;
}
```

```
#include <iostream>
#include <string>
#include <cmath>
using namespace std;

#include "sun.h"
#ifdef _MSC_VER
#define _CRT_SECURE_NO_WARNINGS
#endif
// Function prototype
void sunriseset (double, double, int, int, int, int&, int&);
void adjusttime (int, char, bool, TextTime);
char DigitChar(int);

void sunriseset (double lat, double lon, int mo, int da, int yr,
                 int& RI, int& SE)
{
    double FC,TN,X,Y,T,T9,S,A,Z,YD,WD,pi;
    double MA,A0,A1,ML,L0,L1,C,N,E,EO,W;
    double DL,TL,OB,DE,RA,S0,H,ZT;
//    double Q,TA,RV,JD;
    int sw,sw1;
    void f1(void);
    pi = 3.141592654;
    FC = 2 * pi;OB=0.409095;
    L0=4.88376619;L1=0.017202791;
    A0=6.23471229;A1=0.01720197;
    E =0.016728;
    EO=0.00218;

    TN = (lon/FC) + 0.5;
    X = 1; Y = 1;
    sw = 0;goto f1;
    f1_1:
    T9 = T;
    X = da; Y = mo;
    sw = 1;goto f1;
    f1_2:
    YD=T-T9+1;
    X=floor(T+1)/7;Y=floor(X);
    WD=floor(7*(X-Y)+0.5);
    T=T+3449.5+TN;
    X=YD-WD;
    goto f2;
    f2_1:// Q=ML-RA;
    X=-0.0143;
    goto f4;
    f4_1:
    if (fabs(Y)>=1)
    {
        RI = -9999;
        SE = -9999;
    }
    S0=Z;H=-S0;
    sw1=0;goto f5;
    f5_1:
    X=ZT+EO;
    sw1=0;goto f6;
```

```
f6_1:
RI=int(X*100+Y);
H=S0;
sw1=1;goto f5;
f5_2:
X=ZT+EO;
sw1=1;goto f6;
f6_2:SE=int(X*100+Y);
goto f_end;

// Subroutine f1 (810)
f1:T = 367*(yr-1980);
T=T-floor(7*(yr+floor((Y+9)/12))/4);
if ((Y-9) > 0) S= 1;if((Y-9) < 0) S=-1; if ((Y-9) == 0) S= 0;
A=fabs(Y-9);
Z=floor((yr+S*floor(A/7))/100);
T=T-floor(3*(Z+1)/4);
T=T+floor(275*Y/9)+X - 0.5;
// JD=T+2447689;
if (sw == 0) goto f1_1;
if (sw == 1) goto f1_2;

// Subroutine f2 (900)
f2:MA=A0+A1*T;
ML=L0+L1*T;
X=sin(ML);Y=cos(ML);
sw = 0;goto f3;
f3_1:
ML=Z;
DL=2*E*sin(MA)+1.25*(E*E)*sin(2*MA);
// TA=MA+DL;
TL=ML+DL;
// RV=(1-(E*E))/(1+E*cos(TA));
X=sin(TL)*sin(OB);Y=sqrt(1-(X*X));
sw=1;
goto f3;
f3_2:
DE=Z;if (Z>pi) Z=-FC;
X=sin(TL)*cos(OB);Y=cos(TL);
sw=2;
goto f3;
f3_3:
RA=Z;
goto f2_1;

// Subroutine f3 (570)
f3:C=0;N=0;
if (Y != 0.0)
    Z=X/Y;
else
{
    Z=0;C=1;
    if (X<0) N=1;
}
Z=atan(Z);
if (C==1) Z=pi/2-Z;
if (N==1) Z=-1*Z;
if (Y<0) Z=Z+pi;
```

```
    if (Z<0) Z=Z+2*pi;
    if (sw==0) goto f3_1;
    if (sw==1) goto f3_2;
    if (sw==2) goto f3_3;
    if (sw==3) goto f3_4;
    if (sw==4) goto f3_5;

    // Subroutine f4 (770)
    f4:Y=(X-sin(lat)*sin(DE))/(cos(lat)*cos(DE));
    if (fabs(Y)<=1)
    {
        X=sqrt(1-(Y*Y));
        sw = 3;
        goto f3;
        f3_4:sw=3;
    }
        goto f4_1;

    // Subroutine f5 (710)
    f5:ZT=H+RA+lon-ML-pi;
    X=sin(ZT);Y=cos(ZT);
    sw=4;goto f3;
    f3_5:ZT=Z;
    if (sw1 == 0) goto f5_1;
    if (sw1 == 1) goto f5_2;

    //Subroutine f6 (740)
    f6:W=X*24/FC;X=floor(W);
    Z=(W-X)*60;Y=floor(Z);
    Z=floor((Z-Y)*60);
    if (sw1 == 0) goto f6_1;
    if (sw1 == 1) goto f6_2;

    f_end:Z=1;
}

char DigitChar (int inDig)
{
    char dig = inDig + 48;
    return dig;
}

void adjusttime (int time, char tzone, bool DST, TextTime timeout)
{
    char addchar[2];
    char suffix[3];
    TextTime tempTime;
    strcpy(suffix,"");
    int i,j;
    int adjust = 0;        // Assume UTC - no adjustment
    switch (tzone)
    {
        case 'E':   if (DST)
                        adjust = 400;
                    else
                        adjust = 500;
                    break;
        case 'C':   if (DST)
```

```
                    adjust = 500;
                else
                    adjust = 600;
                break;
    case 'M':   if (DST)
                    adjust = 600;
                else
                    adjust = 700;
                break;
    case 'P':   if (DST)
                    adjust = 700;
                else
                    adjust = 800;
};

time = time - adjust;
if (time < 0)
    time +=2400;
if (tzone == 'U')
    strcpy(suffix,"Z");
else
{
    if (time > 1200)
        strcpy(suffix,"pm");
    else
        strcpy(suffix,"am");
    time %= 1200;
    if (time < 100)
        time += 1200;
}
strcpy(tempTime,"");
strcpy(addchar," ");
if (time > 1000)
{
    addchar[0] = DigitChar(time / 1000);
    strcpy(tempTime,addchar);
}
else if (tzone == 'U')
    strcpy(tempTime,"0");
addchar[0] = DigitChar((time % 1000) / 100);
strcat(tempTime,addchar);
addchar[0] = DigitChar((time % 100) / 10);
strcat(tempTime,addchar);
addchar[0] = DigitChar(time % 10);
strcat(tempTime,addchar);
strcat(tempTime,suffix);

if (tzone != 'U')
{
    if (time > 1000)
        i = 2;
    else
        i = 1;
    j = strlen(tempTime);
    while (j >= i)
    {
        tempTime[j+1] = tempTime[j];
        j--;
```

```
        }
        tempTime[i] = ':';
    }
    strcpy(timeout,tempTime);
}


void Sun_Rise_Set(  double latitude, double longitude,
                    int mon, int day, int year,
                    char timeZone, bool DST,
                    TextTime sun_rise, TextTime sun_set)
{
    int rise,set;

    // Convert to radians
    latitude = latitude * 3.14 / 180;
    longitude = longitude * 3.14 / 180;
    sunriseset(latitude,longitude,mon,day,year,rise,set);

    adjusttime(rise, timeZone, DST, sun_rise);
    adjusttime(set,  timeZone, DST, sun_set );
}
```

Enter a date in the format of yyyymmdd: 20150629

Enter an airport three character code: mbs

SUN 28 JUN 2015 Rise: 5:58am am Set: 9:21pm pm

MON 29 JUN 2015 Rise: 5:58am am Set: 9:21pm pm

TUE 30 JUN 2015 Rise: 5:59am am Set: 9:21pm pm

WED 01 JUL 2015 Rise: 5:59am am Set: 9:21pm pm

THU 02 JUL 2015 Rise: 6:00am am Set: 9:21pm pm

FRI 03 JUL 2015 Rise: 6:00am am Set: 9:21pm pm

SAT 04 JUL 2015 Rise: 6:01am am Set: 9:20pm pm

Do you want to continue? Y/N: y

Enter a date in the format of yyyymmdd: 20151228

Enter an airport three character code: LAX

SUN 27 DEC 2015 Rise: 6:57am am Set: 4:52pm pm

MON 28 DEC 2015 Rise: 6:58am am Set: 4:52pm pm

TUE 29 DEC 2015 Rise: 6:58am am Set: 4:53pm pm

WED 30 DEC 2015 Rise: 6:58am am Set: 4:54pm pm

THU 31 DEC 2015 Rise: 6:58am am Set: 4:54pm pm

FRI 01 JAN 2016 Rise: 6:59am am Set: 4:55pm pm

SAT 02 JAN 2016 Rise: 6:59am am Set: 4:56pm pm

Do you want to continue? Y/N: Y

Enter a date in the format of yyyymmdd: 20151106

Enter an airport three character code: DeN

SUN 01 NOV 2015 Rise: 6:28am am Set: 4:58pm pm

MON 02 NOV 2015 Rise: 6:29am am Set: 4:57pm pm

TUE 03 NOV 2015 Rise: 6:30am am Set: 4:55pm pm

WED 04 NOV 2015 Rise: 6:31am am Set: 4:54pm pm

THU 05 NOV 2015 Rise: 6:33am am Set: 4:53pm pm

FRI 06 NOV 2015 Rise: 6:34am am Set: 4:52pm pm

SAT 07 NOV 2015 Rise: 6:35am am Set: 4:51pm pm

Do you want to continue? Y/N: y

Enter a date in the format of yyyymmdd: 20150704

Enter an airport three character code: LAF

SAT 27 JUN 2015 Rise: 6:19am am Set: 9:22pm pm

SUN 28 JUN 2015 Rise: 6:20am am Set: 9:22pm pm

MON 29 JUN 2015 Rise: 6:20am am Set: 9:22pm pm

TUE 30 JUN 2015 Rise: 6:21am am Set: 9:22pm pm

WED 01 JUL 2015 Rise: 6:21am am Set: 9:22pm pm

THU 02 JUL 2015 Rise: 6:22am am Set: 9:21pm pm

FRI 03 JUL 2015 Rise: 6:22am am Set: 9:21pm pm

Do you want to continue? Y/N:

Enter a date in the format of yyyymmdd: 20151111

Enter an airport three character code: FWA

Invalid Airport Code! Exiting.Press any key to continue . . .

Enter a date in the format of yyyymmdd: 20130931

Enter an airport three character code: dfw

Invalid date! Exiting.Press any key to continue . . .