```
//
// Shaun Chemplavil U08713628
// shaun.chemplavil@gmail.com
// C/C++ Programming III : Intermediate Programming with Objects
// 151116  Raymond L.Mitchell III
// Date.h
// Win10
// Visual C++ 19.0
//
// File contains the Date class definition
//

#ifndef SHAUNCHEMPLAVIL_DATE_H
#define SHAUNCHEMPLAVIL_DATE_H

#include <iostream>
#include <ctime>
using std::cerr;

namespace ShaunChemplavil
{
   class Date
   {
   public:

      inline Date(int month, int day, int year);
      inline Date();
      inline int  getMonth() const;
      inline int  getDay() const;
      inline int  getYear() const;
      // Print Date in "standard" US calendar format
      void display() const;

   private:
      int month, day, year;
   };
}

   // Constructor
   ShaunChemplavil::Date::Date(int month, int day, int year)
   {
      // Default the max number of days equal to 31 (odd months)
      int FebMonth = 2, maxDays = 31;

      //  We need to check if our input was valid
            // Check if day is valid
      if ((this->month != FebMonth) && (bool)(this->month % 2))
         // number of days in an "even" month
         maxDays = 30;
      else // Month is February, we are ignoring leap years
      {
         maxDays = 28;
```

```
      }

      // Check if month, then day, then year is valid, if all valid set vars
      if ((month < 1) || (month > 12))
         cerr << "\nERROR: INVALID MONTH VALUE!\n";
      else if ((day < 1) || (day > maxDays))
         cerr << "\nERROR: INVALID DAY VALUE!\n";
      else if (year < 1)
         cerr << "\nERROR: INVALID YEAR VALUE!\n";
      // Valid Date!
      else
      {
         this->year = year;
         this->day = day;
         this->month = month;
      }

   }

   // Default constructor using current date
   ShaunChemplavil::Date::Date()
   {
      //  Starting year of tm struct
      int tmYearStart = 1900;
      // tm Struct indexes month via 'zero indexing'
      int tmMonthStart = 1;
      time_t currTime = time(0);
      // Converting from time_t to tm structure
      tm *today = localtime(&currTime);

      this->year = today->tm_year + tmYearStart;
      this->month = today->tm_mon + tmMonthStart;
      this->day   = today->tm_mday;

   }

   int ShaunChemplavil::Date::getMonth() const
   {
      return this->month;
   }


   int ShaunChemplavil::Date::getDay() const
   {
      return this->day;
   }

   int ShaunChemplavil::Date::getYear() const
   {
      return this->year;
   }

```

```

#endif

```

```cpp
//
// Shaun Chemplavil U08713628
// shaun.chemplavil@gmail.com
// C/C++ Programming III : Intermediate Programming with Objects
// 151116  Raymond L.Mitchell III
// Date.cpp
// Win10
// Visual C++ 19.0
//
// File contains the display member function for the Date class
//

#include <iostream>
#include "Date.h"
using std::cout;

// Print Date in "standard" US calendar format
void ShaunChemplavil::Date::display() const
{
   cout << getMonth() << "/" << getDay() << "/" << getYear() << "\n";
}

```

```cpp
//
// Shaun Chemplavil U08713628
// shaun.chemplavil@gmail.com
// C/C++ Programming III : Intermediate Programming with Objects
// 151116  Raymond L.Mitchell III
// hw2.cpp
// Win10
// Visual C++ 19.0
//
// Test Program for the Date class
//

#include <iostream>
#include "Date.h"
using std::cout;
using std::cerr;

int main()
{
   // declare original date variables (arbitrary values)
   int month = 6, day = 1, year = 2015;

   // exercise all of Date's public functions
   ShaunChemplavil::Date validDate(month, day, year), todaysDate;

   //exercising the display function
   cout << "The today's date is: ";
   todaysDate.display();

   //exercising the display function
   cout << "The declared date is: ";
   validDate.display();

   // exercising all of the get functions
   cout << "Declared Month is " << validDate.getMonth() << "\n"
      << "Declared Day is " << validDate.getDay() << "\n"
      << "Declared Year is " << validDate.getYear() << "\n";

   // testing setMonth error handling;
   cout << "\n\nTest error handling of invalid Month:\n";
   ShaunChemplavil::Date badMonth(year, day, year);
   cout << "Confirm that garbage is in object\n";
   badMonth.display();

   cout << "\nTest error handling of invalid Day:\n";
   ShaunChemplavil::Date badDay(month, year, year);
   cout << "Confirm that garbage is in object\n";
   badDay.display();

   cout << "\nTest error handling of invalid Year:\n";
   ShaunChemplavil::Date badYear(month, day, -year);
   cout << "Confirm that garbage is in object\n";
```

```
    badYear.display();
}

```