Conor Nee

Note: Lab and Prelab exercises are completed in the Logbook.java file, and Test 3 is activated in TestLogbook.java

**Bridge**

**Test 1**

|  |  |  |  |
| --- | --- | --- | --- |
| **Test case** | **Logbook month** | **No. days in month** | **Checked** |
| Simple month | 1 2002 | 31 | Y |
| Month in past | 7 1998 | 31 | Y |
| Month in the future | 11 2015 | 30 | Y |
| Current month | 5 2015 | 31 | Y |
| Feb (not leap) | 2 1999 | 28 | Y |
| Feb (leap) | 2 2000 | 29 | Y |
| Invalid month | 13 2002 | 31 | Y |

**Test 2**

|  |  |  |  |
| --- | --- | --- | --- |
| **Test case** | **Logbook entries** | **Expected result** | **Checked** |
| Record entries for the first and fifteenth of the month | 2 100  15 200 | 2 100  15 200 | Y |
| Record entries for the first and last day of the month | 1 300  31 500 | 1 300  31 500 | Y |
| Record entries for all the Fridays in the month | 2 140  9 120  16 150  23 200  30 300 | 2 140  9 120  15 150  23 200  30 300 | Y |
| Change the entry for the first day | 1 100  1 300 | 1 100  1 300 | Y |

**Test 3**

|  |  |  |  |
| --- | --- | --- | --- |
| Test case | Logbook month | Expected result | Checked |
| Simple month | 1 2000 | 6 | Y |
| Month in the past | 7 1998 | 3 | Y |
| Month in the future | 12 2008 | 1 | Y |
| Current month | 5 2014 | 5 | Y |
| Feb (non leap) | 2 2002 | 5 | Y |
| Feb (leap) | 2 200 | 2 | Y |

**Postlab A**

private boolean leapYear ( ) // A leap year is a year that is divisible by 4 but not by 400, except when also divisible by 100

{

if (logYear % 4 ! =0)

return false;

else if (logYear % 100 != 0)

return true;

else if (logYear % 400 != 0)

return false;

else

return true;

}

**Postlab B**

**Part B**

In terms of time and space, what is the cost of defining the data member logCalendar to

implement leapYear( )?

A logCalendar object is already instantiated by a Logbook object, so there is no significant cost towards memory. The cost in computation is also insignificant, as the program would have to implement the code in leapYear in the daysInMonth method anyways.

In terms of time and space, what is the cost (or savings) of implementing leapYear( ) without

declaring the GregorianCalendar class data member logCalendar?

Because the default constructor instantiates logCalendar anyways, there is no significant saving in space, as the Logbook object will already have a reference towards logCalendar. The saving/cost in time depends on the efficiency of the two leap year algorithms, but both should have O(1) complexity, so it shouldn’t be very significant.