Project 2 Report

**Notable obstacles**

1. My year calculation was off because I first tried:

yc = y%100/4 + y%100;

cout << "yc is " << yc << endl;

This gave me incorrect answers since division has a higher order of precedence over modulus. To fix this, parentheses were added around y%100 to ensure this is first calculated before dividing by 4.

1. Checking for leap year offset

At first, I tried checking for the leap year offset with just:

**if** (m == "January" && y%4 == 0 && y%100 != 0) {

mc = mc-1;

}

However, this did not work because years evenly divisible by 400, which are leap years, were not accounted for. To fix this, I used nested if statements. The final code checks first if the year is evenly divisible by 4 to be considered a leap year. Then, it checks if that year is also evenly divisible by 100 to not be considered a leap year. Lastly, if that year is also evenly divisible by 400 to be an exception and be considered a leap year.

**if** (m == "January" && y%4 == 0) {

mc = mc-1;

**if** (y%100 == 0) {

mc = 1;

**if** (y%400 ==0) {

mc = mc-1;

}

}

}

**List of test data and reasons**

1. January 1, 2000
   1. See if leap year offset for a year divisible by 400 in the month code calculation is recognized
2. January 1, 2020
   1. See if leap year offset for a year evenly divisible by 4 but not 100 in the month code calculation is recognized
3. February 1, 1800
   1. See if a year that is evenly divisibly by 4 and 100 but not 400 will not be recognized as a leap year in the month code calculation
4. January 1, 2021
   1. See if a year that isn’t a leap year will be recognized
5. March 1, 2000
   1. See if the leap year offset is not applied to a leap year when the month is not January or February
6. January, 1, 0
   1. See if the bounds for the year inputted is enforced
7. jan 1, 2021
   1. See if the acceptable month format for the month inputted is enforced
8. January 0, 2021
   1. See if the bounds for the day inputted is enforced
9. Jan 0, 2020
   1. See if only the first error message is printed when both the month and day inputs are wrong
10. Jan 0, 0
    1. See if only the first error message is printed when the month, day, and year inputs are wrong
11. January 0, 0
    1. See if only the first error message is printed when both the day and year inputs are wrong