**Amy Wang**

**MSCI 240 Fall 2018**

**Instructor: Dr Mark Hancock**

**September 16th, 2018**

**Project 1, Part 1**

**REVISED**

**Introduction:**

The task at hand is to design a test plan and code for a date class that has various methods. These methods provide the class with many useful functions. However, it is unknown if the code is free of errors which is why, for each method, test cases must be made to check the code’s soundness. The test plan should outline all the test cases in plain English and will describe the process of the test and expected result for which the test code will execute with the correct syntax.

**Background Info:**

Date Class Constructors:

* Date(int year, int month, int day): create a custom date instance
* Date(): creates a date with today’s values

Date Class Methods:

* getDay(): returns Date object’s day value
* getMonth(): returns Date object’s month value (1-12)
* getYear(): returns Date object’s year
* getDayOfWeek(): returns Date object’s string date value
* isLeapYear(): returns Boolean of if it is a leap year
* nextDay(): increases the day by 1
* toString(): returns string value of Date object

**Test Plan:**

Part 1 of the test plan will be focused on the constructors to check whether various date objects can be created without errors. First thing is to test if creating today’s date instance is correct. Next will be any valid date, a date with an invalid year, a date with an invalid month, a date with an invalid day, a very high date, February 29th of a leap year, and February 29th of a non-leap year.

Part 1

* 1. Construct a date object for today’s date and print to console, the current date is expected to be printed to the console
  2. Construct a custom date with valid year, month, date then print to console, the entered date is expected to be printed to the console
  3. Construct a custom date with an invalid year, an error is expected to be printed
  4. Construct a custom date with an invalid month, an error is expected to be printed
  5. Construct a custom date with an invalid date, an error is expected to be printed
  6. Construct a custom date with a high date value, should be valid and entered date will be printed
  7. Construct a date with a leap year with February 29, entered date is expected to be printed
  8. Construct a date that’s not a leap year with February 29, an error is expected to be printed

Part 2

Part 2 of the test plan will test each of “get” methods to check if the outputted results are as expected as well as testing logically incorrect data to observe what occurs. The *toString* methods are also tested in this section. One valid date, one high date, the first date possible, and an extremely high date are created to run the methods with.

Test Date 1: 2018/2/14 | Test Date 2: 3000/12/31 (High date)

Test First Date: 1753/1/1

1. Print *getDay* method for test date 1, the expected result is 14
2. Print *getDay* method for test date 2, the expected result is 31
3. Print *getDay* method for the first date, the expected result is 1
4. Print *getMonth* method for test date 1, the expected result is 2
5. Print *getMonth* method for test date 2, the expected result is 12
6. Print *getMonth* method for the first date, the expected result is 1
7. Print *getYear* method for test date 1, the expected result is 2018
8. Print *getYear* method for the first date, the expected result is 1753
9. Print *getYear* method for a high date, the expected result is 3000
10. Print *getDayOfWeek* method for test date 1, the expected result is Wednesday
11. Print *getDayOfWeek* method for test date 2, the expected result is Wednesday
12. Print *getDayOfWeek* method for the first date, the expected result is Monday
13. Print *getDayOfWeek* method for leap year date, the expected result is Monday
14. Print *toString* method for today, the expected result is 2018/9/16
15. Print *toString* method for test date 2, the expected result is 2018/2/14
16. Print *toString* method for first date, the expected result is 1753/1/1

Part 3

Part 3 of the test plan deals with the *nextDay* function and various significant dates. These dates being the last day of the year, the last day of the month, February 28th of a leap year, February 28th of a non-leap year, some arbitrary date, and the earliest date. The method *checkNextDay* will create a Date instance, print that, then call the *nextDay* function and print the Date instance again.

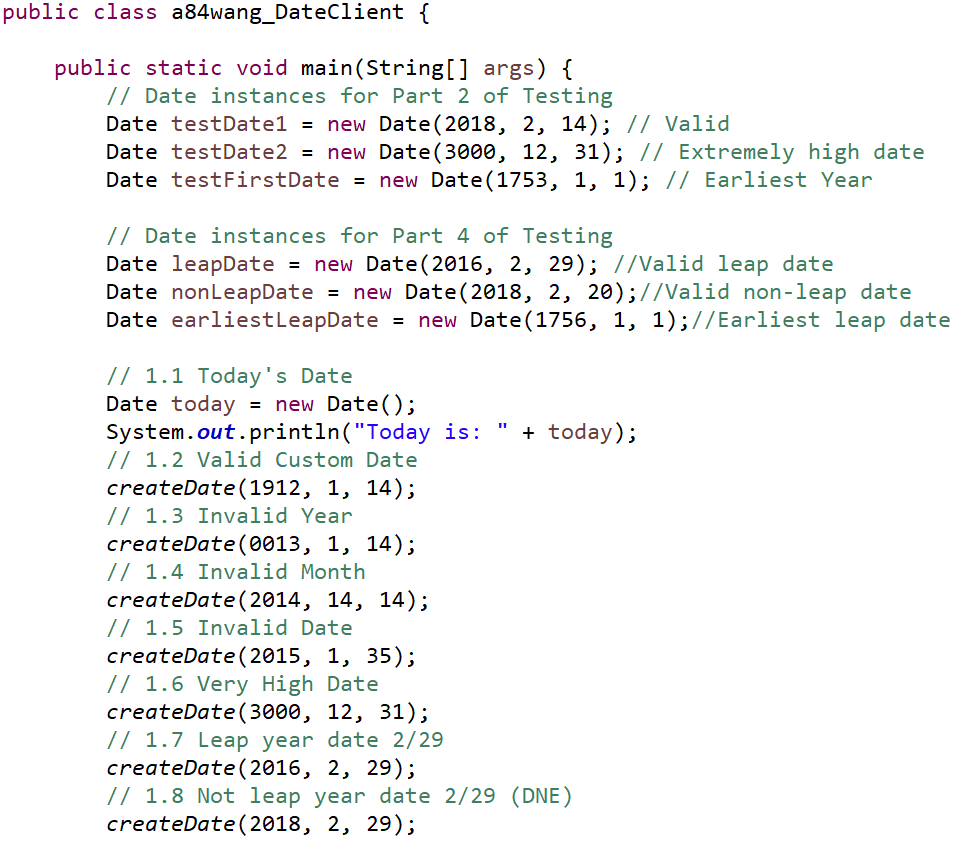
1. Print the next day of some set date (2018/9/16), the expected result is 2018/9/17
2. Print the next day of February 28th of a leap year (2016/2/28), the expected result is 2016/2/29
3. Print the next day of February 28th of a non-leap year (2018/2/28), the expected result is 2018/3/1
4. Print the next day of the day before a new year (2018/12/31), the expected result is 2019/1/1
5. Print the next day of the day before a new month (2018/9/30), the expected result is 2018/10/1
6. Print the next day of the earliest date (1753/1/1), the expected result is 1753/1/2

Part 4

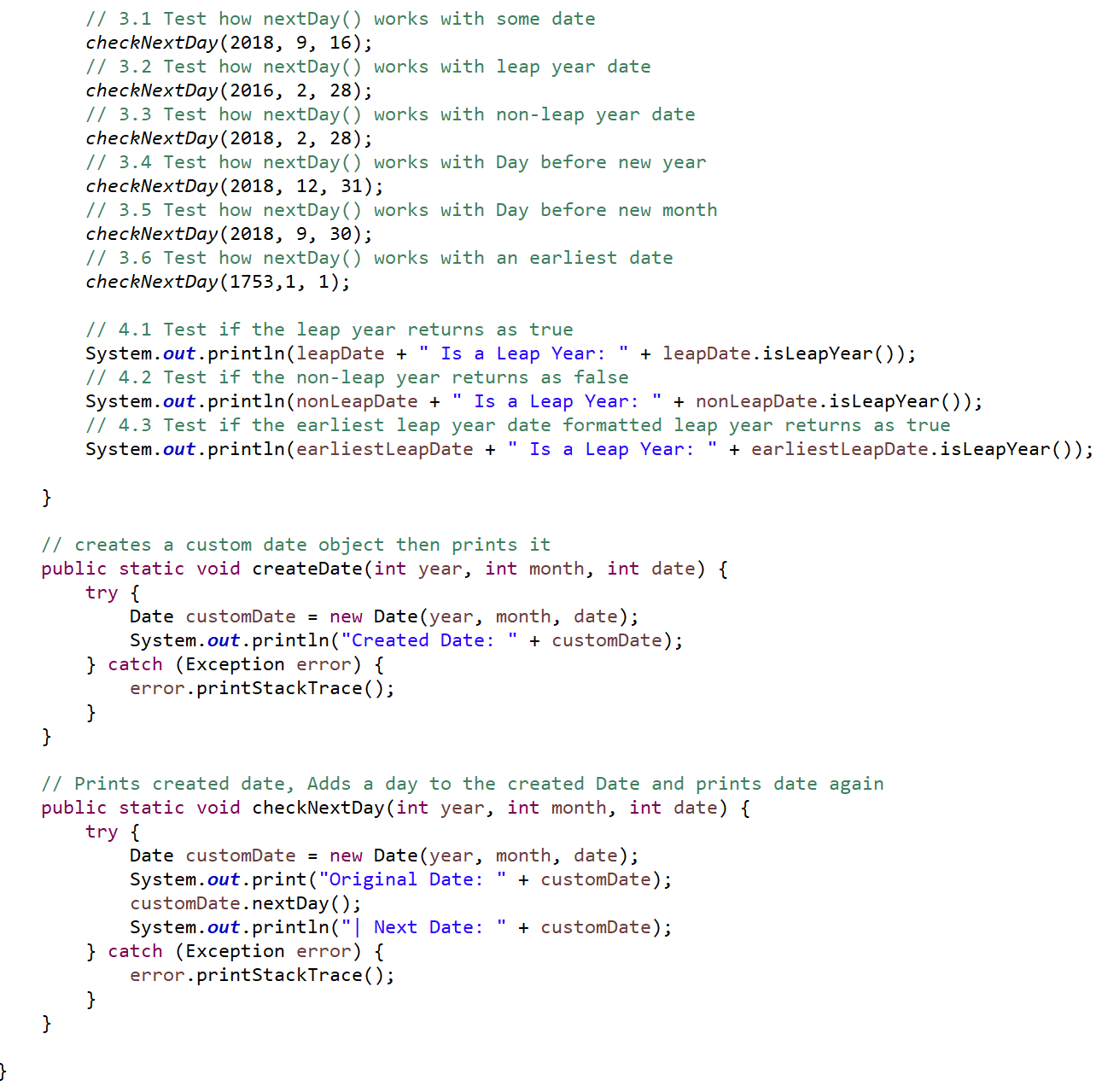
Part 4 tests the *isLeapYear* function to see if it performs as expected. Four dates are created to do just that, a valid date of a leap year, a valid date of a non-leap year, and the earliest leap year date.

1. Print *isLeapYear* method of a valid leap year date (2016/2/29), the expected result is true
2. Print *isLeapYear* method of a valid non-leap year date (2018/2/20), the expected result is false
3. Print *isLeapYear* method of the earliest leap year date (1756/1/1), the expected result is true

**Test Code:**







**Test Console Output:**



# Non-Acknowledgment of Receiving Assistance or Use of Others' Ideas

I received the following help, assistance, or any ideas from classmates, other knowledgeable people, books or non-course websites (please include a description of discussions with the TA or the instructor):

None

# Record of Giving Assistance to Others

I gave the following help, assistance, or ideas to the following classmates (please describe what assistance to whom was given by you):

None

# Declaration

I declare that except for the assistance noted above, assistance provided on the course website, and material provided by the instructor and/or TAs that this is my original work.

I have neither given nor received an electronic or printed version of any part of this code to/from anyone.

I declare that any program output submitted as part of the assignment was generated by the program code submitted and not altered in any way.



Signature: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_