Create a project called `lab4` (if using Visual Studio). Download the file `lab4.cpp` from Blackboard as your starting point.

In `lab4.cpp`, your group will define a class called `Date` that is an abstract data type for a calendar date. The `Date` class will have three member variables to represent the month, day of month, and year, respectively. Your class internally will use the `Month` class provided in the file `lab4.cpp` to represent the month value[1], while the day of month and year values will be of type `int`.

Include all of the following member/friend functions in your `Date` class:

- a default constructor that sets the `Date` to the internal representation for January 1, 2018;
- a value constructor to set the `Date`, using an integer parameter to represent the month value (e.g., `12` for December);
- another value constructor to set the `Date`, using a string parameter containing the first three letters in the name of the month to represent the month value (e.g., "`Dec`" for December);
- a mutator member function that will change the value of the month in a `Date`, given a valid month value as an integer;
- a member function `outputDateAsInt(ostream&)` that writes the `Date` to an output stream passed as an argument to the function, representing the month using an integer (e.g., "`12/31/2018`");
- a member function `outputDateAsString(ostream&)`that writes the Date to an output stream passed as an argument to the function, representing the month using the first three letters in the name of the month (e.g., "`Dec 31, 2018`");
- an overloaded version of the insertion operator `<<` (written as a non-member friend function) that writes the Date given as the right-hand operand of the insertion operator to the output stream given as the left-hand operand of the insertion operator, representing the date using the entire name of the month (e.g., "`December 31, 2018`"); and
- an overloaded version of the pre-increment operator `++` (written as a member function) that retains the same month and day of month as the current Date, but increments the year number.

Write a driver program in the `main` function that will fully test your `Date` class.

1 Note that we are simply defining two classes, `Month` and `Date`, within one compilation unit (the file `lab4.cpp`). We are **not** nesting the `Month` class definition within the `Date` class definition, or vice versa. Although in practice programmers should define each class in a separate compilation unit (with the class interface declared in a header file, and the class implementation in a `.cpp` file), we are using a single compilation unit in these lab assignments in order to simplify and speed up your development process.

The output of your program should look something like this:

```
With the following declarations:
     Date d1, d2(2, 1, 2018), d3("Mar", 1, 2018);
...and using operator<< :
d1 == January 1, 2018
d2 == February 1, 2018
d3 == March 1, 2018

After d3.setMonth(4):
d3 == April 1, 2018

With the following declaration:
     Date d4(12, 31, 2018);
d4.outputDateAsInt(cout) outputs 12/31/2018
d4.outputDateAsString(cout) outputs Dec 31, 2018

++d4 == December 31, 2019
```

When finished, one member of your group should turn in your completed `lab4.cpp` file on Blackboard.