SHOW ALL YOUR WORK. REMEMBER THAT PROGRAM SEGMENTS ARE TO BE WRITTEN IN JAVA.

- Assume that the classes listed in the Java Quick Reference have been imported where appropriate.
- Unless otherwise noted in the question, assume that parameters in method calls are not `null` and that methods are called only when their preconditions are satisfied.
- In writing solutions for each question, you may use any of the accessible methods that are listed in classes defined in that question. Writing significant amounts of code that can be replaced by a call to one of these methods will not receive full credit.

The `APCalendar` class contains methods used to calculate information about a calendar. You will write two methods of the class.

```
public class APCalendar
{
  /** Returns true if year is a leap year and false otherwise */
  private static boolean isLeapYear(int year)
  {  /* implementation not shown */  }


  /** Returns the number of leap years between year1 and year2,
  inclusive.
   *  Precondition: 0 <= year1 <= year2
   */
  public static int numberOfLeapYears(int year1, int year2)
  {  /* to be implemented in part (a) */  }


  /** Returns the value representing the day of the week for the first
  day of year,
   *  where 0 denotes Sunday, 1 denotes Monday, ..., and 6 denotes
   Saturday.
   */
  private static int firstDayOfYear(int year)
  {  /* implementation not shown */  }


  /** Returns n, where month, day, and year specify the nth day of the
  year.
   *  Returns 1 for January 1 (month = 1, day = 1) of any year.
   *  Precondition: The date represented by month, day, year is a valid
   date.
   */
  private static int dayOfYear(int month, int day, int year)
  {  /* implementation not shown */  }


  /** Returns the value representing the day of the week for the given
  date
   *  (month, day, year), where 0 denotes Sunday, 1 denotes Monday, ...,
   *  and 6 denotes Saturday.
```

```
    *  Precondition: The date represented by month, day, year is a valid
    date.
    */
   public static int dayOfWeek(int month, int day, int year)
   {  /* to be implemented in part (b) */  }

   // There may be instance variables, constructors, and other methods
   not shown.

}
```

Write the static method `numberOfLeapYears`, which returns the number of leap years between `year1` and `year2`, inclusive.

In order to calculate this value, a helper method is provided for you.

- `isLeapYear(year)` returns `true` if `year` is a leap year and `false` otherwise.

Complete method `numberOfLeapYears` below. You must use `isLeapYear` appropriately to receive full credit.

```
/** Returns the number of leap years between year1 and year2, inclusive.
 *  Precondition: 0 <= year1 <= year2
 */
public static int numberOfLeapYears(int year1, int year2)
```

----------Part B----------

(b) Write the static method `dayOfWeek`, which returns the integer value representing the day of the week for the given date `(month, day, year)`, where `0` denotes Sunday, `1` denotes Monday, ..., and `6` denotes Saturday. For example, 2019 began on a Tuesday, and January 5 is the fifth day of 2019. As a result, January 5, 2019, fell on a Saturday, and the method call `dayOfWeek(1, 5, 2019)` returns `6`.

As another example, January 10 is the tenth day of 2019. As a result, January 10, 2019, fell on a Thursday, and the method call `dayOfWeek(1, 10, 2019)` returns `4`.

In order to calculate this value, two helper methods are provided for you.

- `firstDayOfYear(year)` returns the integer value representing the day of the week for the first day of `year`, where `0` denotes Sunday, `1` denotes Monday, ..., and `6` denotes Saturday. For example, since 2019 began on a Tuesday, `firstDayOfYear(2019)` returns `2`.
- `dayOfYear(month, day, year)` returns n, where `month, day`, and `year` specify the nth day of the year. For the first day of the year, January 1 `(month = 1, day = 1)`, the value `1` is returned. This method accounts for whether a year is a leap year. For example,
- `dayOfYear(3, 1, 2017)` returns `60`, since 2017 is not a leap year, while
- `dayOfYear(3, 1, 2016)` returns `61`, since 2016 is a leap year.

Class information for this question

```
public class APCalendar
private static boolean isLeapYear(int year)
public static int numberOfLeapYears(int year1, int year2)
private static int firstDayOfYear(int year)
private static int dayOfYear(int month, int day, int year)
public static int dayOfWeek(int month, int day, int year)
```

Complete method `dayOfWeek` below. You must use `firstDayOfYear` and `dayOfYear` appropriately to receive full credit.

```
/** Returns the value representing the day of the week for the given
date
 *  (month, day, year), where 0 denotes Sunday, 1 denotes Monday, ...,
 *  and 6 denotes Saturday.
 *  Precondition: The date represented by month, day, year is a valid
 date.
 */
public static int dayOfWeek(int month, int day, int year)
```