Name _____ Period_____

1. This question involves the use of check digits, which can be used to help detect if an error has occurred when a number is entered or transmitted electronically.

The `CheckDigit` class is shown below. You will write one method of the `CheckDigit` class.

```
public class CheckDigit {

/**
*    Returns the check digit for num
*    Precondition: The number of digits in num is between
*    one and six, inclusive.
*    num >= 0
*/
public static int getCheck(int num)
{ /* implementation not shown */ }

/**
*    Returns true if numWithCheckDigit is valid, or false otherwise,
*    as described in part (a)
*    Precondition: The number of digits in numWithCheckDigit is
*    between two and seven, inclusive.
*    numWithCheckDigit >= 0
*/
public static boolean isValid(int numWithCheckDigit)
{ /* to be implemented in part (a) */ }

// There may be variables and methods not shown.
}
```

(a) Write the `isValid` method. The method returns true if its parameter `numWithCheckDigit`, which represents a number containing a check digit, is valid, and false otherwise. The check digit is always the rightmost digit of `numWithCheckDigit`. The following table shows some examples of the use of `isValid`.

Method Call	Return Value	Explanation
`getCheck(159)`	`2`	The check digit for `159` is `2`.
`isValid(1592)`	`true`	The number `1592` is a valid combination of a number `(159)` and its check digit `(2)`.
`isValid(1593)`	`false`	The number `1593` is not a valid combination of a number `(159)` and its check digit `(3)` because `2` is the check digit for `159`.

Complete method isValid below. You must use getCheck appropriately to receive full credit.

```
/**
 *  Returns true if numWithCheckDigit is valid, or false
 *  otherwise, as described in part (a)
 *  Precondition: The number of digits in numWithCheckDigit is
 *  between two and seven, inclusive. * numWithCheckDigit >= 0
*/
public static boolean isValid(int numWithCheckDigit)
```

/3

Score _____/23

(b) A programmer wants to modify the CheckDigit class to keep track of how many times a call to `isValid` is made with an incorrect check digit. Any time a call to `isValid` is made with an incorrect check digit, the count should be increased by one. The programmer would like to implement this change without making any changes to the signature of the `isValid` method or overloading `isValid`.

Write a description of how you would change the `CheckDigit` class in order to support this modification. Do not write the program code for this change.

Make sure to include the following in your response.

- Identify any new or modified variables or methods.
- Describe, for each new or revised variable or method, how it would change or be implemented, including visibility and type.

/2

Score _____/23

2. 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 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.
}
```

(a) Write the 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)
```

/5

Score _____/23

(b) Write the 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 the 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.

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)
```

/4

Score _____/23

3. This question involves the implementation of a fitness tracking system that is represented by the `StepTracker` class. A `StepTracker` object is created with a parameter that defines the minimum number of steps that must be taken for a day to be considered active. The `StepTracker` class provides a constructor and the following methods.

   - `addDailySteps`, which accumulates information about steps, in readings taken once per day
   - `activeDays`, which returns the number of active days
   - `averageSteps`, which returns the average number of steps per day, calculated by dividing the total number of steps taken by the number of days tracked The following table contains a sample code execution sequence and the corresponding results

Statements and Expressions	Value Returned (blank if no value)	Comment
`StepTracker tr = new StepTracker(10000);`		Days with at least 10,000 steps are considered active. Assume that the parameter is positive.
`tr.activeDays();`	`0`	No data have been recorded yet.
`tr.averageSteps();`	`0.0`	When no step data have been recorded, the `averageSteps` method returns `0.0`.
`tr.addDailySteps(9000);`		This is too few steps for the day to be considered active.
`tr.addDailySteps(5000);`		This is too few steps for the day to be considered active.
`tr.activeDays();`	`0`	No day had at least 10,000 steps.
`tr.averageSteps();`	`7000.0`	The average number of steps per day is (14000 / 2).
`tr.addDailySteps(13000);`		This represents an active day.
`tr.activeDays();`	`1`	Of the three days for which step data were entered, one day had at least 10,000 steps.
`tr.averageSteps();`	`9000.0`	The average number of steps per day is (27000 / 3).
`tr.addDailySteps(23000);`		This represents an active day.
`tr.addDailySteps(1111);`		This is too few steps for the day to be considered active.
`tr.activeDays();`	`2`	Of the five days for which step data were entered, two days had at least 10,000 steps.
`tr.averageSteps();`	`10222.2`	The average number of steps per day is (51111 / 5).

Score _____/23

Write the complete `StepTracker` class, including the constructor and any required instance variables and methods. Your implementation must meet all specifications and conform to the example.

/9

Score _____/23

Score _____/23