

## 2019 AP<sup>®</sup> COMPUTER SCIENCE A FREE-RESPONSE QUESTIONS

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

## 2019 AP<sup>®</sup> COMPUTER SCIENCE A FREE-RESPONSE QUESTIONS

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.

## 2019 AP<sup>®</sup> COMPUTER SCIENCE A FREE-RESPONSE QUESTIONS

3. Many encoded strings contain *delimiters*. A delimiter is a non-empty string that acts as a boundary between different parts of a larger string. The delimiters involved in this question occur in pairs that must be *balanced*, with each pair having an open delimiter and a close delimiter. There will be only one type of delimiter for each string. The following are examples of delimiters.

### Example 1

Expressions in mathematics use open parentheses " ( " and close parentheses " ) " as delimiters. For each open parenthesis, there must be a matching close parenthesis.

$(x + y) * 5$  is a valid mathematical expression.

$(x + (y)$  is NOT a valid mathematical expression because there are more open delimiters than close delimiters.

### Example 2

HTML uses `<B>` and `</B>` as delimiters. For each open delimiter `<B>`, there must be a matching close delimiter `</B>`.

`<B> Make this text bold </B>` is valid HTML.

`<B> Make this text bold </UB>` is NOT valid HTML because there is one open delimiter and no matching close delimiter.

# AP<sup>®</sup> COMPUTER SCIENCE A

## 2019 SCORING GUIDELINES

### Question 2: Step Tracker

<b>Class:</b> <code>StepTracker</code>	<b>9 points</b>
--	-----------------

**Intent:** *Define implementation of a class to record fitness data*

- +1**    Declares all appropriate `private` instance variables
- +2**    Constructor
  - +1**    Declares header: `public StepTracker(int ____)`
  - +1**    Uses parameter and appropriate values to initialize instance variables
- +3**    `addDailySteps` method
  - +1**    Declares header: `public void addDailySteps(int ____)`
  - +1**    Identifies active days and increments count
  - +1**    Updates other instance variables appropriately
- +1**    `activeDays` method
  - +1**    Declares and implements `public int activeDays()`
- +2**    `averageSteps` method
  - +1**    Declares header: `public double averageSteps()`
  - +1**    Returns calculated `double` average number of steps

# AP<sup>®</sup> COMPUTER SCIENCE A 2019 SCORING GUIDELINES

## Question 2: Scoring Notes

Class <code>StepTracker</code>			9 points
Points	Rubric Criteria	Responses earn the point even if they...	Responses will not earn the point if they...
+1	Declares all appropriate <code>private</code> instance variables		<ul style="list-style-type: none"> <li>omit keyword <code>private</code></li> <li>declare variables outside the class</li> </ul>
+2	Constructor		
+1	Declares header: <code>public StepTracker(int ____)</code>	<ul style="list-style-type: none"> <li>omit keyword <code>public</code></li> </ul>	<ul style="list-style-type: none"> <li>declare method <code>private</code></li> </ul>
+1	Uses parameter and appropriate values to initialize instance variables	<ul style="list-style-type: none"> <li>initialize primitive instance variables to default values when declared</li> </ul>	<ul style="list-style-type: none"> <li>fail to use the parameter to initialize some instance variable</li> <li>fail to declare instance variables</li> <li>initialize local variables instead of instance variables</li> <li>assign variables to parameters</li> </ul>
+3	<code>addDailySteps</code> method		
+1	Declares header: <code>public void addDailySteps(int ____)</code>	<ul style="list-style-type: none"> <li>omit keyword <code>public</code></li> </ul>	<ul style="list-style-type: none"> <li>declare method <code>private</code></li> </ul>
+1	Identifies active days and increments count	<ul style="list-style-type: none"> <li>put valid comparison erroneously in some other method</li> </ul>	<ul style="list-style-type: none"> <li>fail to use the parameter as part of the comparison</li> <li>fail to increment a count of active days</li> <li>fail to increment an instance variable</li> <li>compare parameter to some numeric constant</li> </ul>
+1	Updates other instance variables appropriately		<ul style="list-style-type: none"> <li>update another instance variable only on active days</li> <li>update another instance variable inappropriately</li> <li>fail to update appropriate instance variable</li> <li>update a local variable</li> </ul>
+1	<code>activeDays</code> method		
+1	Declares and implements <code>public int activeDays()</code>	<ul style="list-style-type: none"> <li>return appropriate count of active days where the instance variables were updated improperly in <code>addDailySteps</code> or <code>activeDays</code></li> </ul>	<ul style="list-style-type: none"> <li>declare method <code>private</code></li> <li>return value that is not the number of active days</li> <li>fail to return a value</li> </ul>

# AP<sup>®</sup> COMPUTER SCIENCE A

## 2019 SCORING GUIDELINES

### Question 2: Scoring Notes (continued)

Points	Rubric Criteria	Responses earn the point even if they...	Responses will not earn the point if they...
+2	averageSteps method		
+1	Declares header: <code>public double averageSteps()</code>	<ul style="list-style-type: none"> <li>omit keyword <code>public</code></li> </ul>	<ul style="list-style-type: none"> <li>declare method <code>private</code></li> </ul>
+1	Returns calculated <code>double average</code> number of steps	<ul style="list-style-type: none"> <li>maintain instance variables improperly but calculate appropriate average</li> <li>fail to handle the special case where no days are tracked</li> </ul>	<ul style="list-style-type: none"> <li>use integer division</li> <li>calculate something other than steps divided by days</li> <li>fail to return</li> </ul>

# AP<sup>®</sup> COMPUTER SCIENCE A

## 2019 SCORING GUIDELINES

### Question 2: Step Tracker

```
public class StepTracker
{
    private int minSteps;
    private int totalSteps;
    private int numDays;
    private int numActiveDays;

    public StepTracker(int threshold)
    {
        minSteps = threshold;
        totalSteps = 0;
        numDays = 0;
        numActiveDays = 0;
    }

    public void addDailySteps(int steps)
    {
        totalSteps += steps;
        numDays++;
        if (steps >= minSteps)
        {
            numActiveDays++;
        }
    }

    public int activeDays()
    {
        return numActiveDays;
    }

    public double averageSteps()
    {
        if (numDays == 0)
        {
            return 0.0;
        }
        else
        {
            return (double) totalSteps / numDays;
        }
    }
}
```

These canonical solutions serve an expository role, depicting general approaches to solution. Each reflects only one instance from the infinite set of valid solutions. The solutions are presented in a coding style chosen to enhance readability and facilitate understanding.