Past-paper walk-through

Class Variables and Methods ▪ 3.7

eXercise

Questions in this set

- 2019 Q2

Reading the mark scheme

- **B1** a mark that stands on its own – the point is either made or not.
- **M1** a *method* mark: the right approach, even if the number is wrong.
- **A1** an *answer* mark, and it usually needs its M mark first.
- **C1** a working mark on the way to the answer.
- // separates answers the examiner accepts equally.
- **ecf** = error carried forward: a later part is marked on your own earlier answer.
- **owtte** = "or words to that effect" – the wording need not match.

Question 2

2019 ▪ Q2

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

Question 2

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).

Question 2

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.

Question 2

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.
[illegible]		This is too few steps for the day to be considered

Question 2

`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).

Solution 2

- Model solution for the full StepTracker class:
- public class StepTracker {
- private int minSteps;
- private int totalSteps;
- private int numDays;
- private int numActiveDays;
- public StepTracker(int threshold) {
- minSteps = threshold;

Solution 2

- totalSteps = 0;
- numDays = 0;
- numActiveDays = 0;
- }
- public void addDailySteps(int steps) {
- totalSteps += steps;
- numDays++;
- if (steps >= minSteps) { numActiveDays++; }
- }

Solution 2

- public int activeDays() {
- return numActiveDays;
- }
- public double averageSteps() {
- if (numDays == 0) { return 0.0; }
- else { return (double) totalSteps / numDays; }
- }
- }

Solution 2

- How the 9 points are earned: +1 declares all appropriate private instance variables (e.g. minSteps, totalSteps, numDays, numActiveDays – must be private and inside the class); Constructor (+2 total): +1 declares the header public StepTracker(int ____), +1 uses the parameter and appropriate values to initialize every instance variable (failing to initialize one, or assigning parameters to themselves instead of instance variables, forfeits this point); addDailySteps method (+3 total): +1 declares the header public void addDailySteps(int ____), +1 identifies active days (compares the parameter against the threshold) and increments an active-day count, +1 updates the other instance variables appropriately (e.g. running total and day count) on every call, not just active days; activeDays method (+1 total): declares and implements public int activeDays() returning the correct count, given correct updates elsewhere;

Solution 2

averageSteps method (+2 total): +1 declares the header public double averageSteps(), +1 returns the correctly calculated double average of steps per day, including handling the special case of zero tracked days (e.g. returning 0.0) without failing – using integer division instead of double division forfeits this point.