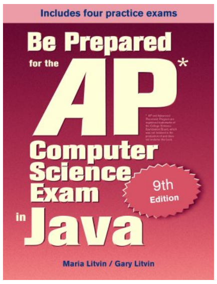

Ninth Edition

Be Prepared
for the
AP
Computer Science Exam in Java

Chapter 5: Annotated Solutions to Past Free-Response Questions

2019

Maria Litvin
Phillips Academy, Andover, Massachusetts

Gary Litvin
Skylight Publishing, Andover, Massachusetts

Skylight Publishing
Andover, Massachusetts

ISBN 978-0-9972528-3-5

Skylight Publishing
9 Bartlet Street, Suite 70
Andover, MA 01810

web: www.skylit.com
e-mail: sales@skylit.com
support@skylit.com

www.skylit.com/beprepared/x2019all.zip contains complete Java code, including solutions and test programs for runnable projects.

The free-response questions for this exam are posted on apstudent.collegeboard.org and, for teachers, on AP Central:

- For students: `apstudents.collegeboard.org`
- For teachers: `apcentral.collegeboard.org`

Scoring guidelines for teachers are usually posted over the summer.

Question 1

Part (a)

```
public static int numberOfLeapYears(int year1, int year2)
{
  int count = 0;

  for (int year = year1; year <= year2; year++)
    if (isLeapYear(year)) [1]
      count++;

  return count;
}
```

Notes:

1. You will not receive full credit if you code your own `isLeapYear` rather than calling the provided private helper method.

Part (b)

```
public static int dayOfWeek(int month, int day, int year)
{
  return (firstDayOfYear(year) +
         dayOfYear(month, day, year) - 1) % 7; [1]
}
```

Notes:

1. Same here: must call the provided helper methods.

Question 2

```
public class StepTracker
{
  private int minSteps, totalSteps; [1]
  private int totalDays, actDays;

  public StepTracker(int goal)
  {
    minSteps = goal;
    totalSteps = 0; [2]
    totalDays = 0;
    actDays = 0;
  }

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

  public int activeDays()
  {
    return actDays;
  }

  public double averageSteps()
  {
    if (totalDays == 0) [3]
      return 0.0;
    else
      return (double)totalSteps / totalDays; [4]
  }
}
```

Notes:

1. Don't be tempted to introduce an array or an `ArrayList` in this question; just keep track of the numbers. Arrays and/or `ArrayList` will be tested in Question 3.

2. This and the next two statements are optional, because instance variables are automatically initialized to default values: 0 for `int`s, 0.0 for `double`s, `false` for `boolean`s, `null` for objects. It is helpful to remember that elements of an array created using the `new` operator are also automatically initialized to default values. But you must explicitly initialize local variables before they are used — they do not get default values.

3. A special case: avoid division by 0.

4. Need a cast to `double` before division. Alternatively, declare `totalSteps` a `double`.

Question 3

Part (a)

```
public ArrayList<String> getDelimitersList(String[] tokens)
 {
   ArrayList<String> dels = new ArrayList<String>();

   for (String t : tokens)
     if (t.equals(openDel) || t.equals(closeDel)) [1]
       dels.add(t);

   return dels;
 }
```

Notes:

1. Always use `equals` when comparing strings, not ==.

Part (b)

```
public boolean isBalanced(ArrayList<String> delimiters)
{
  int count = 0; [1]

  for (String del : delimiters)
  {
    if (del.equals(openDel))
      count++;
    else
    {
      count--;
      if (count < 0)
        return false; [2]
    }
  }
  return count == 0; [3]
}
```

Notes:

1. No need for two separate counts.

2. You cannot add

```
else
  return true;
```

here, of course.

3. Recall that you can return the value of a Boolean expression. This is better than more verbose

```
if <condition>
  return true;
else
  return false;
```

Question 4

Part (a)

```
public LightBoard(int numRows, int numCols)
{
  lights = new boolean[numRows][numCols];

  for (int r = 0; r < numRows; r++)
    for (int c = 0; c < numCols; c++)
      lights[r][c] = Math.random() < 0.4; [1, 2]
}
```

Notes:

1. You can assign the value of a Boolean expression to a Boolean variable. This is a bit shorter than

```
   if (Math.random() < 0.4)
      lights[r][c] = true;
    else
      lights[r][c] = false;
```

2. Or `<= 0.4`

Part (b)

```
public boolean evaluateLight(int row, int col)
{
  int count = 0;

  for (int r = 0; r < lights.length; r++) [1]
    if (lights[r][col])
      count++;

  if (lights[row][col] && count % 2 == 0)
    return false; [2]

  if (!lights[row][col] && count % 3 == 0)
    return true;

  return lights[row][col];
}
```

Notes:

1. First compute the number of lights that are "on" in the column, then decide what to do with it.

2.

```
else
  return true;
```

here would be a mistake.