



Ninth Edition

Be Prepared
for the
AP
Computer Science
Exam in Java

Chapter 5: Annotated Solutions
to Past Free-Response Questions

2025

Maria Litvin

Phillips Academy, Andover, Massachusetts

Gary Litvin

Skylight Publishing, Andover, Massachusetts

Skylight Publishing
Andover, Massachusetts

**Copyright © 2026 by
Maria Litvin, Gary Litvin, and Skylight Publishing**

All rights reserved. No part of this publication may be reproduced, stored in a retrieval system, or transmitted, in any form or by any means, electronic, mechanical, photocopying, recording, or otherwise, without the prior written permission of the authors and Skylight Publishing.

ISBN 978-0-9972528-3-5

Skylight Publishing
14 Lincoln Street
Andover, MA 01810

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

www.skylit.com/beprepared/x2025all.zip contains complete Java code, including solutions and test programs for runnable projects.

The free-response questions for this exam are posted on AP Central:
<https://apcentral.collegeboard.org/courses/ap-computer-science-a/exam/past-exam-questions>

Scoring guidelines for teachers are usually posted over the summer.

Question 1

Part (a)

```
public int walkDogs(int hour)
{
    int numWalked = Math.min(maxDogs,
                             company.numAvailableDogs(hour));1
    company.updateDogs(hour, numWalked);
    return numWalked;
}
```

Notes:

1. Or, to stay within the Quick Reference:

```
int numWalked = company.numAvailableDogs(hour);
if (numWalked > maxDogs)
    numWalked = maxDogs;
```

Part (b)

```
public int dogWalkShift(int startHour, int endHour)
{
    int totalPay = 0;

    for (int hour = startHour; hour <= endHour; hour++)
    {
        int numWalked = walkDogs(hour);
        totalPay += 5*numWalked;
        if (numWalked == maxDogs || (hour >= 9 && hour <= 17))1
            totalPay += 3;
    }
    return totalPay;
}
```

Notes:

1. The set of parentheses around the && expression is optional.

Question 2

```
public class SignedText
{
    private String firstName, lastName;

    public SignedText(String first, String last)
    {
        firstName = first;
        lastName = last;
    }

    public String getSignature()
    {
        if (firstName.length() == 0) 1
            return lastName;
        else
            return firstName.substring(0, 1) + "-" + lastName; 2
    } 3

    public String addSignature(String text)
    {
        String sig = getSignature();

        int pos = text.indexOf(sig);
        if (pos == -1)
            return text + sig;
        else if (pos > 0) 4
            return text;
        else
            return text.substring(sig.length()) + sig;
    }
}
```

Notes:

1. Or:
 `if (firstName.equals(""))`
2. `firstName.charAt(0)` instead of `firstName.substring(0, 1)` would earn full credit.
3. In an alternative solution, the signature would be calculated in the constructor and saved in an instance variable; the `getSignature` method would just return that value.
4. The question states that the signature occurs at the beginning or at the end of `text` — it cannot be in the middle.

Question 3

Part (a)

```
public Round(String[] names)
{
    competitorList = new ArrayList<Competitor>();

    for (int i = 0; i < names.length; i++)
        competitorList.add(new Competitor(names[i], i+1));
}
```

Part (b)

```
public ArrayList<Match> buildMatches()
{
    ArrayList<Match> matches = new ArrayList<Match>();

    int i = 0;
    if (competitorList.size() % 2 == 1)
        i = 1; 1

    int j = competitorList.size() - 1;

    while (i < j)
    {
        matches.add(new Match(competitorList.get(i),
                               competitorList.get(j)));
        i++;
        j--;
    }

    return matches;
}
```

Notes:

1. Or:

```
int i = competitorList.size() % 2;
```

Question 4

Part (a)

```
public SumOrSameGame(int numRows, int numCols)
{
    puzzle = new int[numRows][numCols];

    for (int r = 0; r < numRows; r++)
    {
        for (int c = 0; c < numCols; c++)
        {
            puzzle[r][c] = (int) (9*Math.random()) + 1;
        }
    }
}
```

Part (b)

```
public boolean clearPair(int row, int col)
{
    for (int r = row; r < puzzle.length; r++)
    {
        for (int c = 0; c < puzzle[0].length; c++)
        {
            if ((r != row || c != col) ^
                && (puzzle[row][col] == puzzle[r][c] ||
                    puzzle[row][col] + puzzle[r][c] == 10))
            {
                puzzle[row][col] = 0;
                puzzle[r][c] = 0;
                return true;
            }
        }
    }
    return false;
}
```

Notes:

1. It is important not to compare the element of `puzzle` to itself.